Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Т.Д. Карминская, Е.Н. Рыбалка

БАЗЫ ДАННЫХ

Учебное методическое пособие

2013

Корректор: Афанасьева Г.А.

Карминская Т.Д., Рыбалка Е.Н.
Базы данных: учебное методическое пособие. — Томск: Факультет дистанционного обучения, ТУСУР, 2013. — 46 с.

Методическое пособие содержит описание указаний по выполнению лабораторных и контрольных работ по курсам информационного обеспечения систем автоматизированного проектирования и систем управления. Приведены примеры программных файлов по темам лабораторных работ.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью изучения курса «Базы данных» является получение основных навыков ведения информационного фонда, проектирования баз данных, изучение методов манипулирования данными.

Для изучения программной системы управления базами данных СУБД FoxPro студентам необходимо выполнить текстовую контрольную работу и лабораторные работы, которые описаны в данном методическом пособии.

1 ПРОГРАММА ЛЕКЦИОННОГО КУРСА

1.1 Основные понятия. Информация и данные. Понятия информационного обеспечения. Специфика данных САПР. Состав информационного фонда САПР.

1.2 Способы ведения информационного фонда. Аспекты представления данных. Организация информационного обеспечения на основе концепции баз данных.

1.3 Архитектура систем баз данных. Уровни представления данных.

1.4 Модели данных. Трехуровневая архитектура представления.

1.5 Стандарты СУБД. Принципы разработки СУБД. Обобщенное определение СУБД.

1.6 Представление данных. Основные определения.

1.7 Виды взаимосвязей между данными.

1.8 Основные модели данных.

1.9 Иерархические модели данных. Описание и терминология.

1.10 Сетевые системы. Отличия иерархий от сети.

1.11 Реляционные модели. Реляционные СУБД.

1.12 Базисные средства манипулирования реляционными данными.

1.13 Реляционная алгебра.

1.14 Реляционное исчисление.

1.15 Проектирование баз данных.

1.16 Функциональные зависимости.

1.17 Теория нормальных форм. Первая нормальная форма (1НФ). Вторая нормальная форма (2НФ). Третья нормальная форма (3НФ). Четвертая нормальная форма (4НФ).

1.18 Языковые средства СУБД. Языки описания данных.

1.19 Язык запросов SQL.

1.20 Язык запросов QBE.

1.21 Диалоговая система СУБД FoxPro.

2 СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Дейт К. Дж. Введение в системы баз данных : пер. с англ / К. Дж. Дейт. — 6-е изд. — Киев-Москва : Диалектика, 1998. — 784 с. : ил.

2. Хансен Г., Хансен Дж. Базы данных: разработка и управление : пер. с англ. / Г. Хансен, Дж. Хансен. — М. : ЗАО «Издательство БИНОМ», 1999. — 704 с. : ил.

3. Попов А. А. Программирование в среде СУБД FoxPro 2.0. Построение систем обработки данных / А. А. Попов. — М. : Радио и связь, 1993. — 352 с. : ил.

4. Бемер С. FoxPro 2.5. для Windows.: пер. с нем. / С. Бемер.— К. : Торгово-издательское бюро BHV, 1994. — 416 с. : ил.

3 РЕКОМЕНДАЦИИ ПО СОСТАВЛЕНИЮ ПРОГРАММ

Для изучения команд рекомендуется их выполнение в командном окне СУБД FoxPro (оно обычно высвечивается в правом нижнем углу экрана) и просмотр результатов их действия на экране. Возвращаемое функцией значение можно в командном окне вывести на экран командой:

?<функция>.

При отсутствии достаточной литературы по СУБД FoxPro можно изучать команды и функции с использованием встроенной системы помощи FoxPro. Для этого необходимо инициализировать меню HELP, выбрать нужную команду и просмотреть пример ее использования.

Любой пример можно запомнить в программном файле. Для этого необходимо выделить блок примера клавишами перемещения курсора (удерживая нажатой клавишу SHIFT) и занести (скопировать) его в буфер, нажав комбинацию клавиш CTRL+C. Затем, открыв уже имеющийся или создав новый программный файл (командой Modify command), переместить в командный файл содержимое буфера клавишами Ctrl+V.

Для контроля открытых файлов базы данных, открытых индексных файлах, главного индекса и для вывода значений установок на экран можно выполнить команду:

DISPLAY STATUS

Для пояснения действий команд в программный файл можно включать комментарии после двойного символа &&.

Например:

```
Select a
Use Name.dbf in a && инициализация файла базы данных Name.dbf в рабочей области a.
```

Если строка начинается с символа *, тогда все, что следует за этим символом в строке также считается комментарием (этот символ в начале команды, также можно использовать для временного исключения строки с командой из выполнения программы).

4 ЛАБОРАТОРНЫЕ РАБОТЫ

Для выполнения лабораторных работ необходимо установить систему FoxPro и изучить основные понятия и возможности диалоговой среды, описанные в учебном пособии в разделе 7.

Выбор варианта (предметной области) лабораторных работ, а также текстовой контрольной работы осуществляется по общим правилам с использованием следующей формулы:

$$V = (N*K) \text{ div } 100,$$

где V — искомый номер варианта,
N — общее количество вариантов,
div — целочисленное деление,
при V= 0 выбирается максимальный вариант,
K — значение 2-х последних цифр пароля.
Перечень предметных областей представлен в Приложении А.

4.1 Лабораторная работа № 1

Лабораторная работа № 1 состоит из двух заданий: задания № 1 «Создание и модификация файла базы данных» и задания № 2 «Разработка программ локализации и поиска записей». Результатом выполнения лабораторной работы № 1 является выполнение обоих заданий.

Задание № 1. Создание и модификация файла базы данных

Целью работы является получение навыков описания логической структуры файлов базы данных, изучение режимов редактирования и приемов упорядочивания данных, создание и манипулирование индексными файлами.

Порядок выполнения работы

4.1.1 Выбрать опцию New в вертикальном меню File полосы главного меню экранного интерфейса. В ответ появляется область диалога (рис. 4.1) со списком типов файлов, которые можно создать в текущий момент времени. По умолчанию считается выбранным тип Database, т. е. dbf-файл. Это нам и нужно, поэтому активизируем ОК, после чего система переходит в окно диалога (рис. 4.2), поддерживающего процесс создания описания структуры dbf-файла, пока со стандартным именем Untitled (безымянный). Фактически запущена команда Create в чем можно убедиться посмотрев команду в командном окне, т.к. любое действие пользователя, выбранное с помощью инициализации пункта меню, реализуется определенной командой системы и отражается в командном окне.

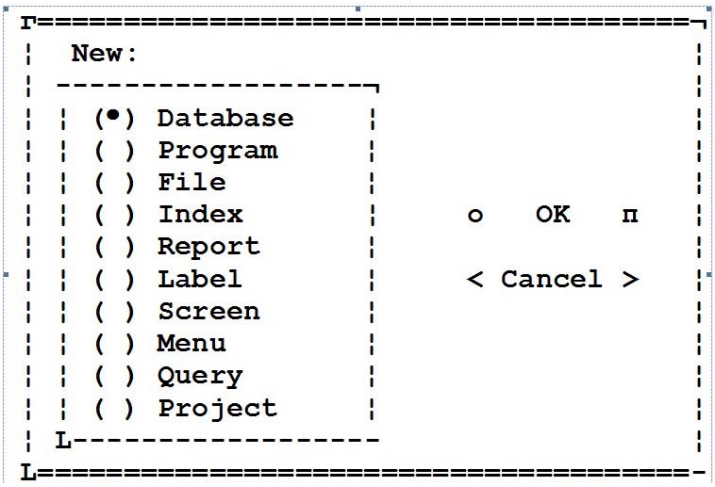


Рис. 4.1 — Выбор типа файла

Пользователю в окне STRUCTURE (рис. 4.2) предлагается для каждого типа данных последовательно, начиная с первого, ввести параметры, которые характеризуют данные:

- *Name-имя данного (длина до 10 символов),*
- *Type — тип,*
- *Width — размер(длину поля)*
- *Dec — точность (имеет смысл только для числовых (Numeric) значений данных и назначает число знаков после запятой).*

Начиная с этого момента, приводимые окна диалога будут отображать действие по работе с конкретными файлами.

Проиллюстрируем процесс создания файла ONE.dbf с полями: CODE,FULL_NAME,ORGANIZ,ADDRESS,NAME_RUK, NUMBER_TLG,CH_MEST,SEASON,CH_PAC_Y.

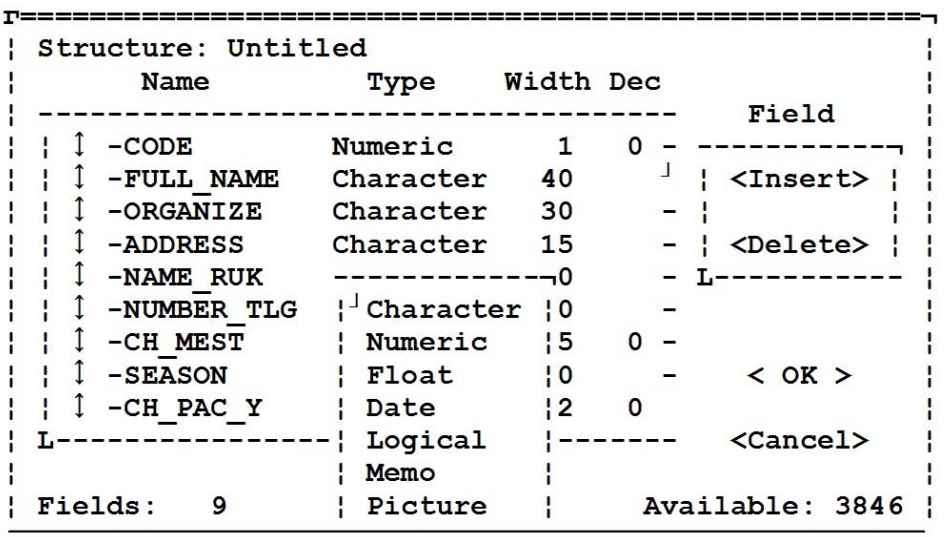


Рис. 4.2 — Создание и описания структуры dbf-файла

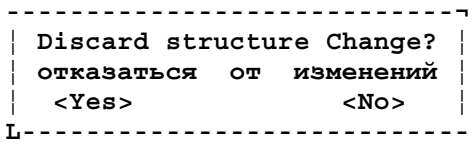
Меню Type (назначение типов данных) вызывается нажатием клавиши SPACE(пробел) после ввода имени поля, когда становится активным поле TYPE. Поле точность (DEC) имеет смысл, а потому и доступно толь-

ко для числовых данных, а поле размер (WIDTH) автоматически устанавливается для логических — 1, дат — 8 (мм/дд/гг) и Мемо полей — 10.

По умолчанию для символьных и числовых данных размер поля установлен равным 10, пользователь, при необходимости, может его скорректировать.

4.1.2 Чтобы вставить описание нового поля в уже существующий список полей, необходимо подвести курсор на описание поля, непосредственно за которым следует вставить новое поле, а затем с помощью клавиши Tab перейти к кнопке INSERT и активизировать ее. В описании появится строка с именем NEW FIELD. Пользователь записывает необходимое имя вместо зарезервированного по умолчанию New field, а также задает его тип и размер.

Аналогично, для удаления описания некоторого поля необходимо вначале установить на него курсор, а затем перейти с помощью клавиши Tab на кнопку DELETE и активизировать ее. Если пользователь намерен отказаться от создаваемого описания полностью, он должен активизировать кнопку CANCEL, после чего система запрашивает



Если выбрать кнопку No — происходит возврат в окно ввода структурных характеристик. Если выбрать кнопку Yes — отмена всех созданных описаний и выход в начальное окно интерфейса FoxPro. В командном окне остается команда CREATE UNTITLED, ее можно вновь запустить, установив на нее курсор и нажав клавишу ENTER.

При выборе кнопки ОК (нормальное завершение формирования описания структуры файла) система переходит в окно (рис. 4.3) определения имени файла (по умолчанию система присваивает имя Untitled).

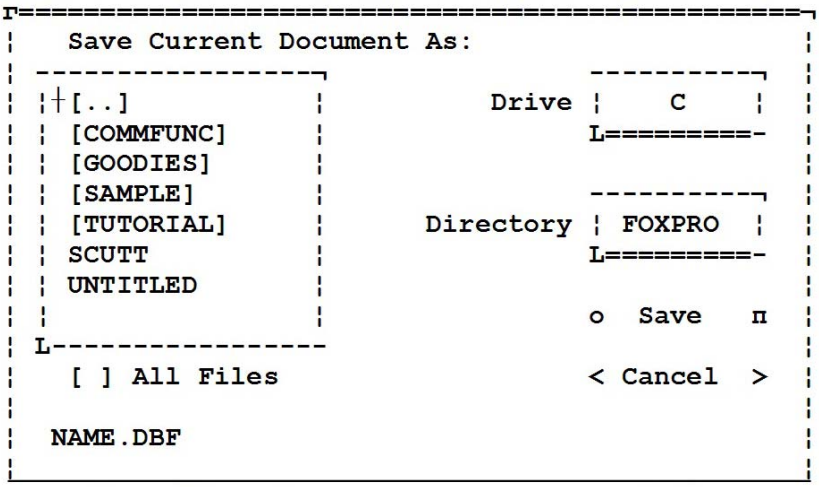


Рис. 4.3 — Формирование имени файла

Для размещения создаваемого файла пользователь может выбрать любое устройство, активизировав переключатель Drive, директорию, активизировав переключатель Directory, и указать (ввести с клавиатуры) имя файла в текстовом окне ввода имени вместо Untitled.

Пользователь может отказаться от всех установок, выбрав кнопку Cancel, либо, выбрав кнопку Save, записать созданный файл (пока пустой) в заданную директорию. В последнем случае (по Save) система спросит не намерен ли пользователь вводить значения данных (заполнить файл базы данных конкретными значениями данных).

```
┆ Input data record now?┆
┆ <<Yes>>         <No>  ┆
```

Если пользователь отвечает <No> — возврат в окно описания структуры файла, по <<Yes>> система переходит в окно создания и редактирования записи, что соответствует запуску команды Append (данная команда отражается в окне Command).

Окно создания-редактирования записей в режиме Append (рис. 4.4) содержит специальную, анкетную форму представления записи в виде вертикально расположенных пар «имя данного — окно для значения данного», причем курсор установлен в окне ввода значения первого данного.

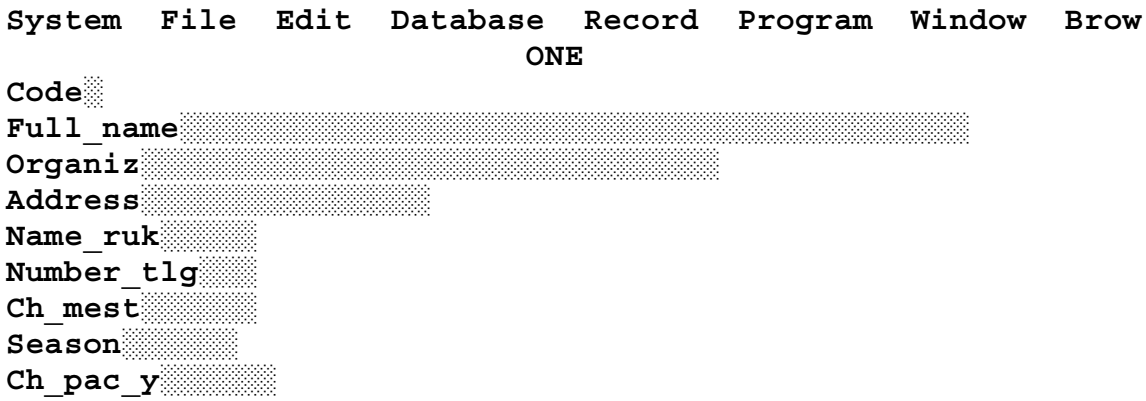


Рис. 4.4 — Ввод данных в режиме Append (Change)

Необходимые значения данных вводятся с клавиатуры. Если конкретное значение данного меньше установленного в описании, необходимо нажать ENTER или стрелку вниз для перехода к вводу значения следующего данного, если размер точно соответствует установленному, то переход осуществится автоматически после заполнения окна значением.

Если число полей велико (не умещается в окне) происходит автоматическая прокрутка при завершении ввода последнего видимого в окне данного (очередное данное «подтягивается» снизу). Можно и листать, просматривать сформированные значения данных в записи, используя стандартные клавиши навигации (PgDn, PgUp, Home, End), в том числе для то-

го, чтобы проверить результаты текущего ввода и, возможно, исправления неточностей.

После завершения ввода очередной записи система предоставляет новую пустую запись. Завершение ввода новых записей осуществляеся нажатием клавиш Ctrl + End.

Для работы с данными пользователь может перейти к табличной форме (рис. 4.5) отображения файла (режим Browse).

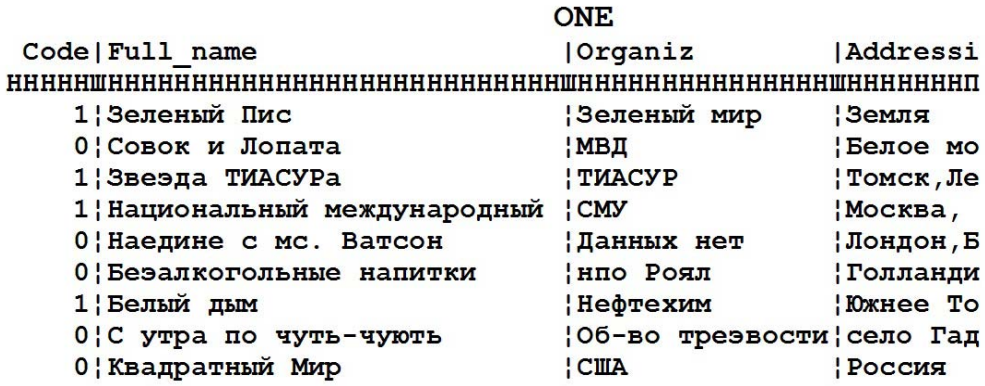


Рис. 4.5 — Окно Browse просмотра, создания и редактирования файлов

Для удаления записей необходимо пометить их на удаление с помощью клавиш Ctrl+T (слева появляется отметка), а физическое удаление записей из файла произойдет только после активизации опции (команды) PACK в меню Database.

4.1.3 ВВОД НОВЫХ ЗАПИСЕЙ:

– открыть требуемый файл данных, выбрав опцию OPEN в вертикальном меню File и отработать диалоговое окно выбора файла выбором типа файла (Database) и имени файла;

– выбрать опцию APPEND (добавить) в вертикальном меню Record с последующим вводом значений данных точно в такой же форме и с теми же правилами, что были приведены в предыдущем разделе (создание нового dbf-файла). Допускается внесение изменений в записи, которые были сформированы в текущем сеансе работы, путем установки на требуемые записи клавишей (PgUp, PgDn) и их модификации.

4.1.4 РЕДАКТИРОВАНИЕ ЗНАЧЕИЙ ДАННЫХ ЗАПИСЕЙ ФАЙЛА

Осуществляется после открытия требуемого файла и последующего выбора режима Change или Browse вертикального меню Record. Совпадающие по форме отображения команды Append и Change отличаются тем, что в Append для корректировки доступны лишь те записи, которые были введены в текущем сеансе работы, а по Change — любые записи файла.

4.1.5 ДОБАВЛЕНИЕ ИНФОРМАЦИИ В ОТКРЫТЫЙ DBF-ФАЙЛ ИЗ ДРУГОГО ФАЙЛА

Осуществляется путем выбора опции Append From из вертикального меню Database с последующим выбором второго файла через последовательные активации (рис. 4.6) опции <From>, затем выбор типа файла в появившемся меню (Database — dbf-файл, Delemited Wiln — текстовый файл с разделителями полей символами: табуляцией (Tab); запятыми (commas); Space (пробел), SDF-текстовой файл жесткого формата, когда одноименные данные в разных записях расположены на одних и тех же местах и имеют одни и те же размеры, затем выбор требуемого файла посредством поиска по дискам и директориям по стандартной схеме, рассмотренной ранее (рис. 4.3).

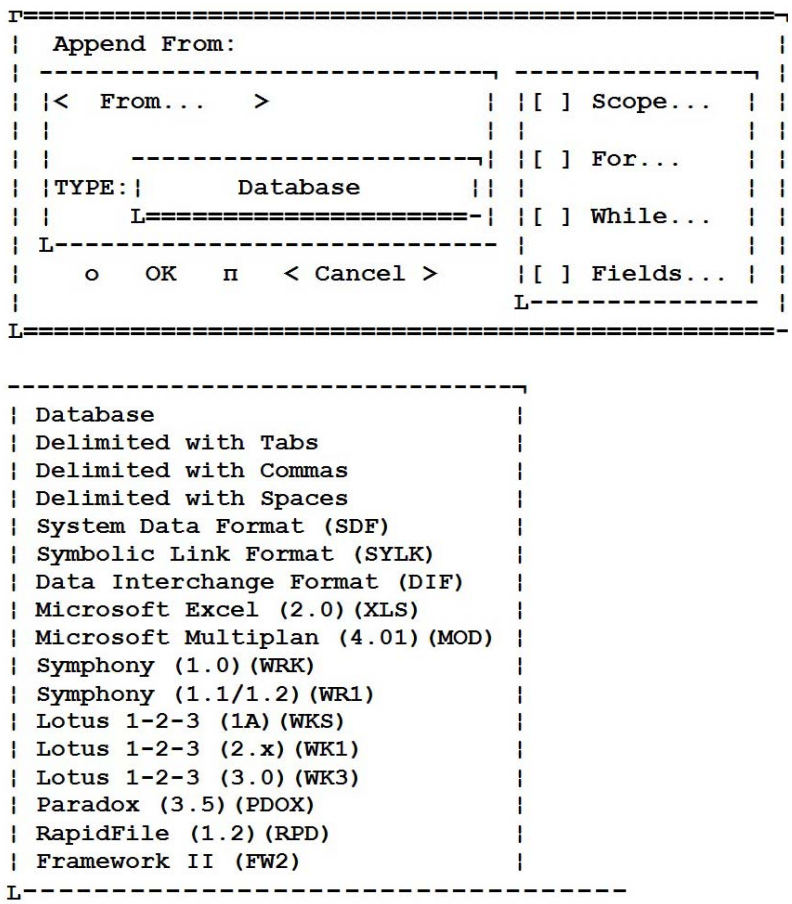


Рис. 4.6 — Окно диалога Append From, выбор типа файла

Окно диалога Append from содержит также стандартные переключатели установки области действия (Scope, For, While) и переключатель Fields, позволяющий отобрать в записи добавляемые к исходному файлу лишь часть полей из записей второго файла. Завершение процесса стандартными клавишами <OK> или <Cancel>.

4.1.6 КОПИРОВАНИЕ ИНФОРМАЦИИ В ФАЙЛ

Инициализируется выбором опции Copy to вертикального меню Database и последующего диалога. Общий вид окна диалога Copy to на рис. 4.7. При активизации Scope можно задать область действия команды, For и While — установить условия отбора (на рис. 4.8 пример формирования выражения для For), Fields — выбрать поля исходного файла, затем выбрать тип формируемого файла, активизируя переключатель TYPE и выбирая в списке как на рис. 4.6), задать имя формируемого файла, активизируя Save as, а затем по стандартной схеме завершить копирование созданием файла выбором <<OK>> или отказом <Cancel>.

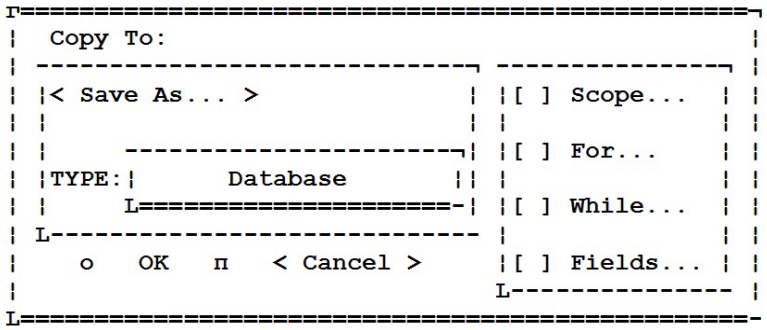


Рис. 4.7 — Окно диалога Copy to

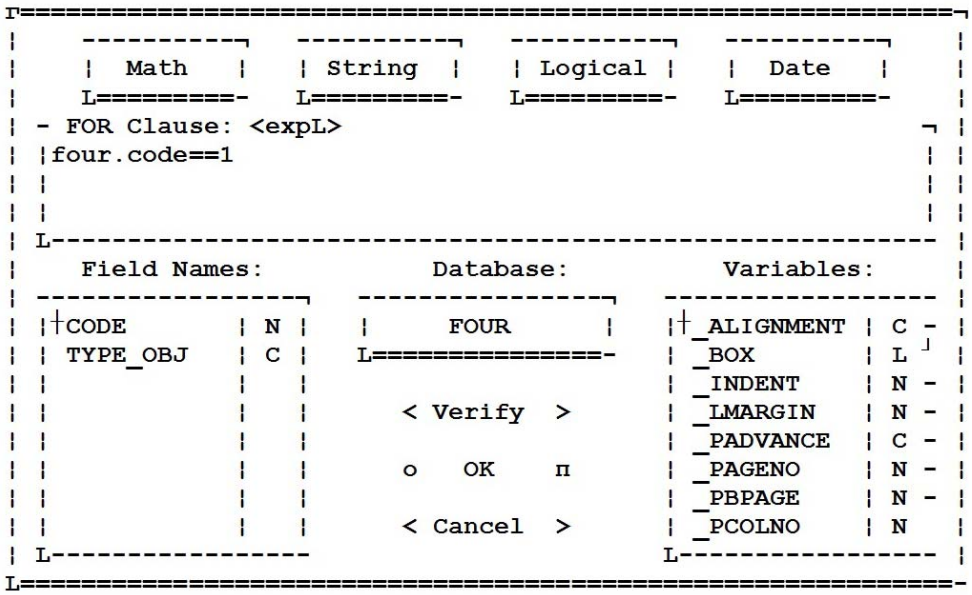


Рис. 4.8 — Формирование выражения для For

4.1.7 РАБОТА В РЕЖИМЕ BROWSE

Режим просмотра и полноэкранного редактирования Browse часто используется, и для этого режима имеется специальное меню настройки и дополнительных функций (оно появляется на линейке главного меню после выбора команды Browse):

Change — инициализация режима отображения записи в режиме шаблона.

Grid off — удаляет вертикальные разделительные линии между столбцами таблицы.

Size field — назначение активного поля (на нем установлен курсор) столбца для последующего изменения размера столбца таблицы с помощью клавиш навигации (стрелка влево или вправо) и после получения требуемого размера — зафиксировать нажатием клавиши ENTER. Если ошибочно выбрано не то поле, можно с помощью клавиши Tab переместиться на другие поля. Необходимо иметь ввиду, что изменяется размер столбца в таблице для отображения поля на экране, но не размер поля в записи. Поэтому при размере столбца меньшем, чем размер записи, значения поля отображается не полностью.

Move Field — выделяет активное поле-столбец для его перемещения в другое место таблицы путем перемещения с помощью стрелок влево/вправо и установки на новое место нажатием клавиши ENTER. Аналогично предыдущей опции перемещение столбца не влечет перемещения поля в структуре записи.

Toggle Delete — помечает (снимает отметку при повторном выборе) активную запись на удаление в виде точки перед записью. Физическое удаление произойдет только после выбора опции PACK в вертикальном меню Database. Аналогично опции Toggle Delete действует комбинация клавиш Ctrl+T.

Append Record — добавляет пустую запись в конце файла (аналогичное действие клавиши Ctrl+R).

Resize Partitions — активизирует разделитель окна (мерцающий курсор), перемещает его с помощью стрелок влево/вправо в место желаемого разделения окна Browse на 2 части, завершение клавишей ENTER.

Unlink Partitions (Link Partitions) — взаимозаменяемые опции.

Если окно просмотра разделено, то можно просматривать файл в каждом подокне, независимо друг от друга (Unlink) либо синхронно (Link) одним курсором-строкой.

Change Partiton — меняет состояние активности между подокнами. Все действия, которые производит пользователь, осуществляются только в активном подокне. Активное подокно выделено более ярким свечением рамки. В различных рабочих окнах могут быть установлены разные формы просмотра (Change, Browse). Во время редактирования полей в окнах просмотра доступны все функции текстового редактора FoxPro.

ВСТАВКА ЗАПИСИ внутрь файла осуществляется после установки курсором на запись, вслед за которой необходимо вставить новую запись путем активизации переключателя INSERT.

ВЫЧИСЛЯЕМЫЕ ПОЛЯ-СТОЛБЦЫ в составе таблицы Browse можно создать с помощью команды Browse Fields, которая должна быть сформирована явно в командном окне. Например, BROWSE FIELDS FIO, DAYPAY, KOLDAY. OPLATA=DAYPAY KOLDAY.

4.1.8 УПОРЯДОЧЕНИЕ ЗАПИСЕЙ В ФАЙЛЕ

При пополнении dbf-файлов новыми записями они автоматически добавляются в конец файла. В то же время, для обеспечения эффективной обработки целесообразно иметь упорядоченную последовательность доступа к записям. Целесобразно определять порядок следования записей в соответствии с возрастанием или убыванием значений данных, входящих в состав ключа, или условий, используемых для группировки записей.

В системе FoxPro может быть установлена физическая и/или логическая упорядоченность записей в dbf-файле.

При ФИЗИЧЕСКОЙ УПОРЯДОЧЕННОСТИ записи файла в памяти следуют друг за другом, в соответствии с возрастанием (убыванием) значения ключа сортировки, который может состоять из одного или нескольких полей (с указанием старшинства полей в ключе). Поэтому при физической сортировке записи перемещаются в памяти, т. е. файл перезаписывается.

Для проведения физической сортировки данных необходимо при открытом файле в вертикальном меню Database выбрать опцию Sort и далее в окне диалога сортировки (рис. 4.9):

– определить поля ключа сортировки, выделяя поля из Database Fields в Sort Order по стандартной схеме Move-Remove в порядке их старшинства;

– установить порядок сортировки (по возрастанию Ascending или по убыванию Descending значения ключа сортировки). Старшинство значений полей определяется старшинством двоичных кодов символов в значении поля (согласно таблице ASCII) и порядком их расположения слева-направо;

– установить область действия Score и условия отбора записей (For. While), если это необходимо. То есть в отсортированный файл можно поместить не все записи исходного;

– осуществить, если это необходимо, отбор полей исходного файла для их размещения в выходном, т. е. в выходной файл можно поместить не все поля записей исходного (стандартная схема рис. 4.10);

– сформировать путь доступа и имя файла в текстовом блоке непосредственно вводом с клавиатуры или используя стандартное окно определения имени файла, используя кнопку Save as;

– отказаться от различия в прописных и строчных буквах путем активизации переключателя Ignore Case.

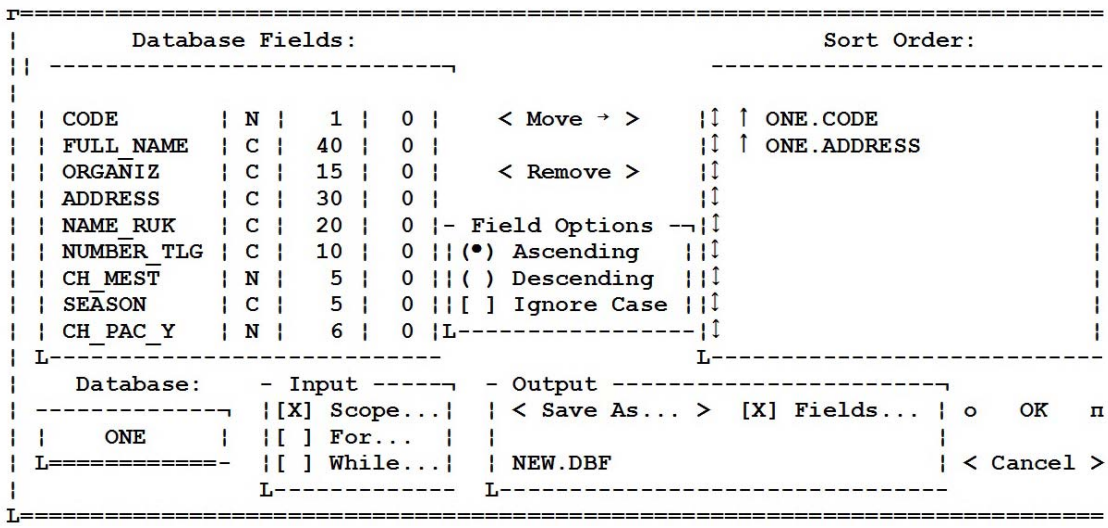


Рис. 4.9 — Окно диалога сортировки

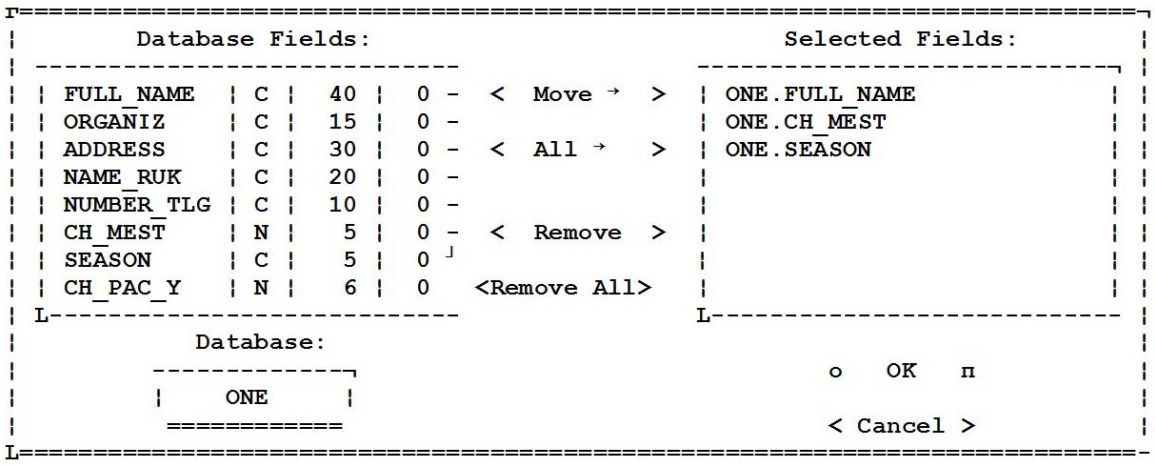


Рис. 4.10 — Отбор полей SEASON, FULL_NAME, CH_MEST
из файла ONE для их размещения в отсортированном файле

ЛОГИЧЕСКАЯ упорядоченность записей осуществляется без изменения физического порядка хранения записей (если не было физической сортировки, то это в порядке ввода записей), за счет создания дополнительного индекса файла, записи которого содержат поля сортировки каждой записи основного (индексируемого) файла и адресуют ссылку на физическую запись основного файла с такими же значениями полей сортировки. Записи индексного файла физически упорядочены по значениям полей сортировки, поэтому с использованием такого индексного файла достаточно просто обеспечивается последовательный доступ к записям основного файла в соответствии с установленным порядком либо быстро определяется адрес основного файла по значению индекса (по значению данных, входящих в ключ сортировки).

Для одного основного файла может быть создано много индексных, и в зависимости от того, какой индексный файл в текущий момент времени открыт вместе с основным и объявлен главным, и будет осуществляться порядок доступа к записям основного файла.

Для объявления индексного файла главным необходимо в окне установки (рис. 4.11) (инициализируется после выбора опции Setup пункта DATABASE главного меню) выбрать <Set Order> и в появившемся списке индексных файлов выбрать необходимый. Для снятия главного индекса — повторить его выбор.

Индексный файл может быть создан путем переработки основного файла либо динамически модифицироваться, если в поля основного файла, используемые для индексирования, вносятся изменения или в основной файл добавляются новые записи. Для обеспечения такой автоматической модификации индексного файла он должен быть открыт во время коррекции соответствующего основного. Таким образом, одновременно может быть открыто до 21 индексного файла для одного основного (dbf) файла, но только один из них (специально помеченный в текущий момент как order) определяет порядок доступа к записям основного файла.

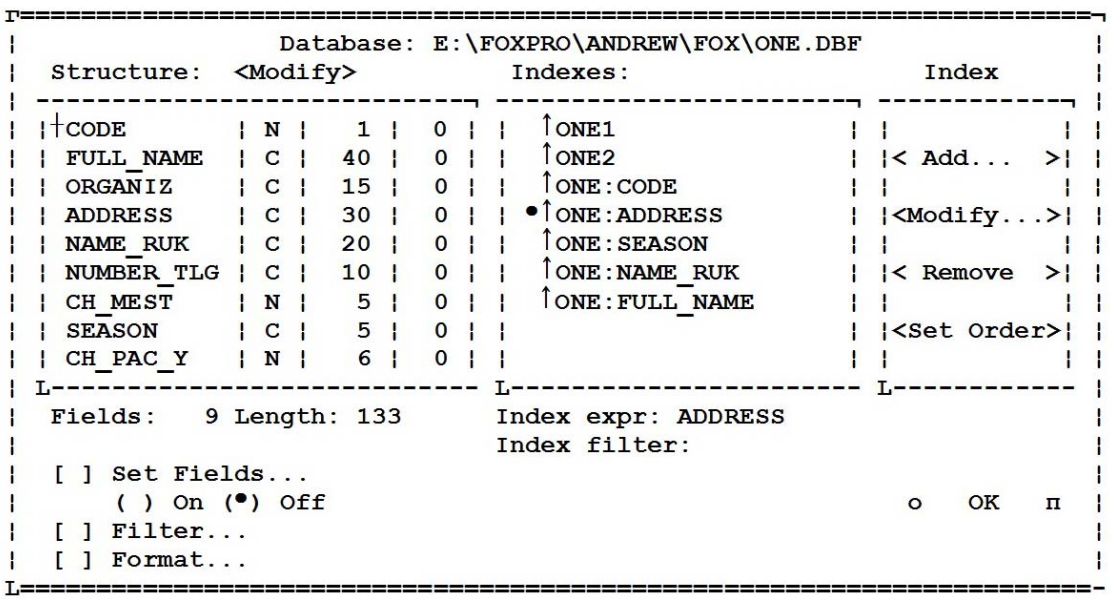


Рис. 4.11 — Окно установки индексов

ОТКРЫТИЕ ИДЕКСНОГО ФАЙЛА осуществляется либо путем выбора Index в списке типов файлов, появляющеся при выборе опции OPEN вертикального меню File, либо при выборе опции Add в окне диалога по установке параметров текущей рабочей области, появляющемся при выборе опции Set up вертикального меню Database (рис. 4.11). В обоих случаях появляется окно открытия индексных файлов (рис. 4.12).

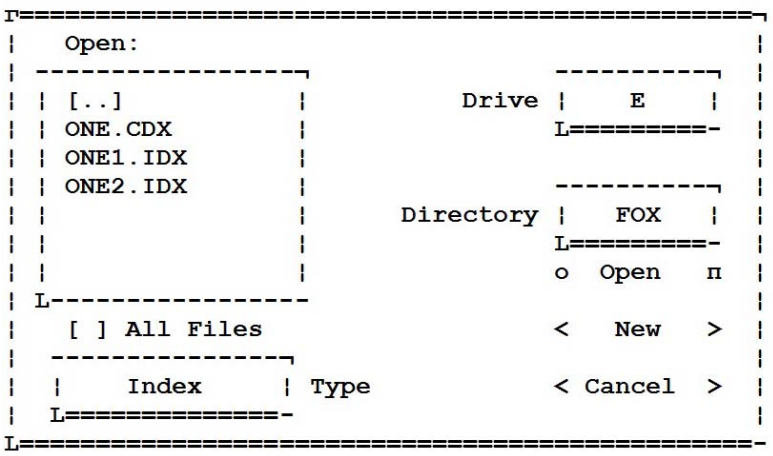


Рис. 4.12 — Окно открытие индексных файлов

Процедура открытия состоит в выборе индексного файла в списке индексных файлов, созданных для открытого dbf-файла. Возможно, придется «искать» индексные файлы по дискам (Drive) и директориям (Directory), хотя не рекомендуется хранить dbf-файлы и созданные для них индексные файлы в разных директориях и, тем более, дисках.
Если необходимо открыть несколько файлов, то процедура повторяется. Активизируя переключатель All Files, можно открыть все индексные файлы сразу.

4.1.9 СОЗДАНИЕ ИДЕКСНОГО ФАЙЛА

В окне-списке появляются лишь индексные файлы, созданные ранее для открытого dbf-файла, если список пуст (как, например, на рис. 4.11), это значит индексные файлы для соответствующего dbf-файла еще не созданы, а выбирая опцию Open, можно перейти к окну диалога по созданию индексного файла.

Необходимо вначале открыть dbf-файл, для которого будет создаваться индексный (опция Open, выбор типа файла — Database, выбор-поиск файла на диске и в дирректории), а затем либо выбрать опцию New, затем Index, либо выбрать Set up, затем активизировать переключатель <Add> в Index и далее аналогичный диалог — New и Index.

Появляется окно диалога «Index on» (рис. 4.13), в котором необходимо либо ввести с клавиатуры индексное выражение (состав индексного ключа, в том числе используя возможность выбора имен полей из списка и их переноса в формируемое выражение), либо, активизировав переключатель <Expr>, перейти в окно построителя выражения, общая схема которого была рассмотрена ранее и иллюстрируется на рис. 4.14, 4.15, при активизации переключателя <Expr> и с использованием меню построителя (рис. 4.16).

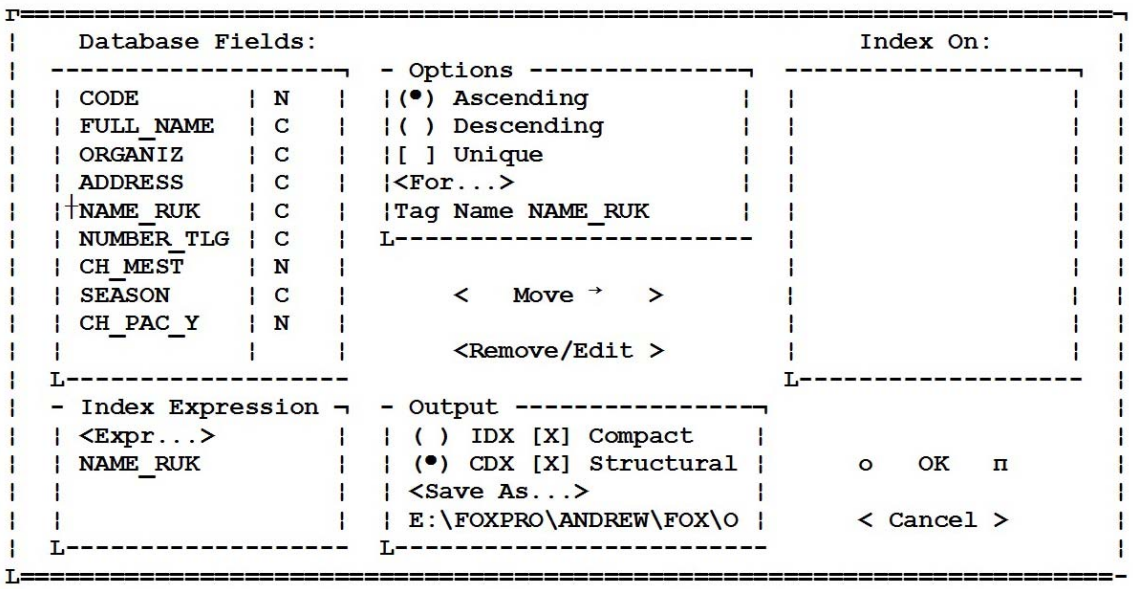


Рис. 4.13 — Окно диалога Index on

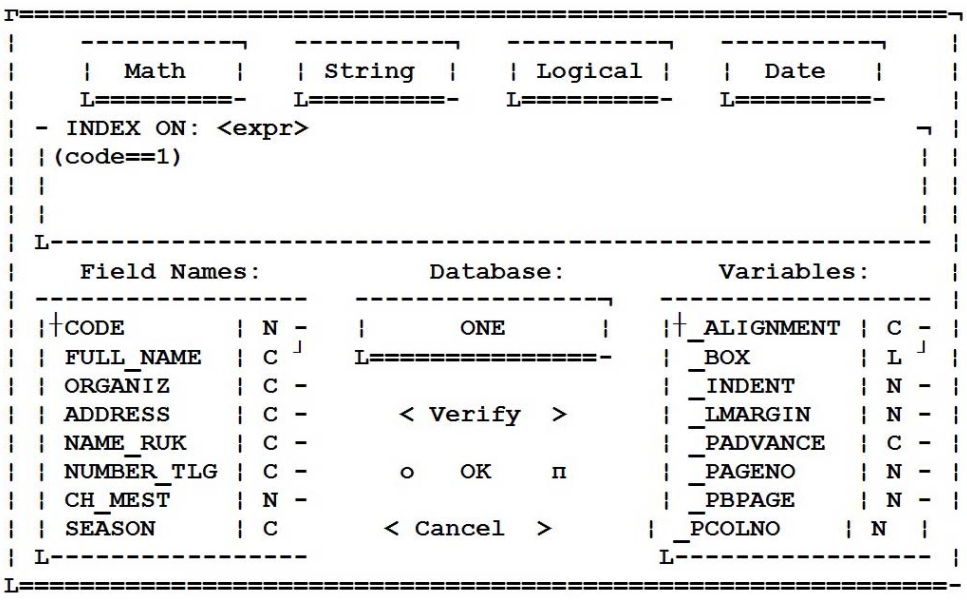


Рис. 4.14 — Окно построителя выражения

Следует заметить, что в подавляющем большинстве случаев выражение имеет тип String (символьный) и заключается в соединении (операция + или — над строковыми переменными) значений полей сортировки в единый ключ.

Использование данных не символьного типа требует использования функций по их преобразованию к символьному виду.

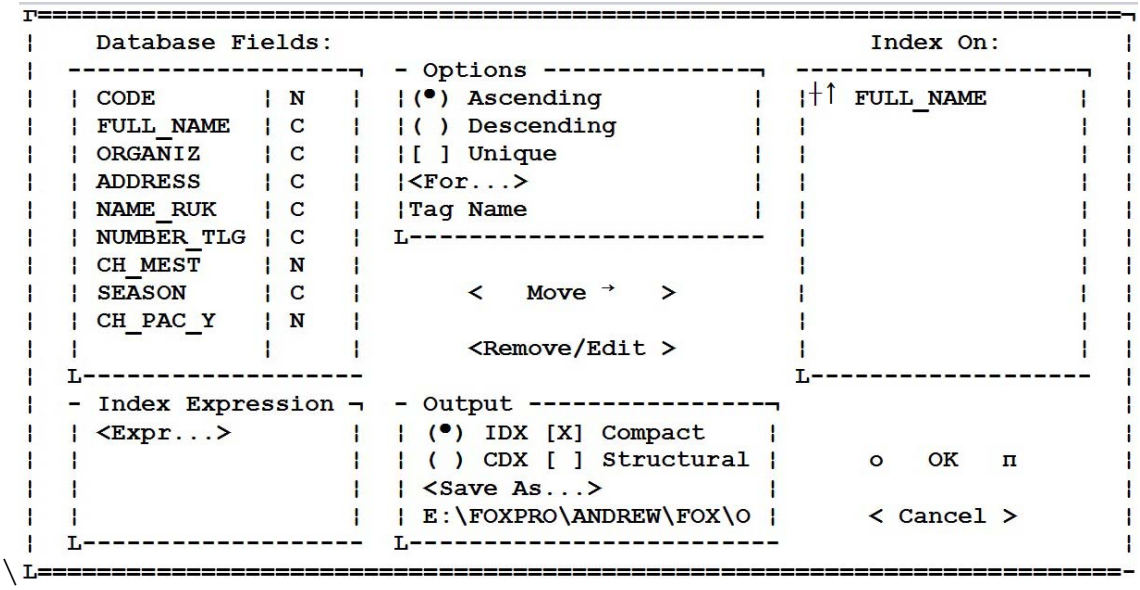


Рис. 4.15 — Диалог формирования индекса

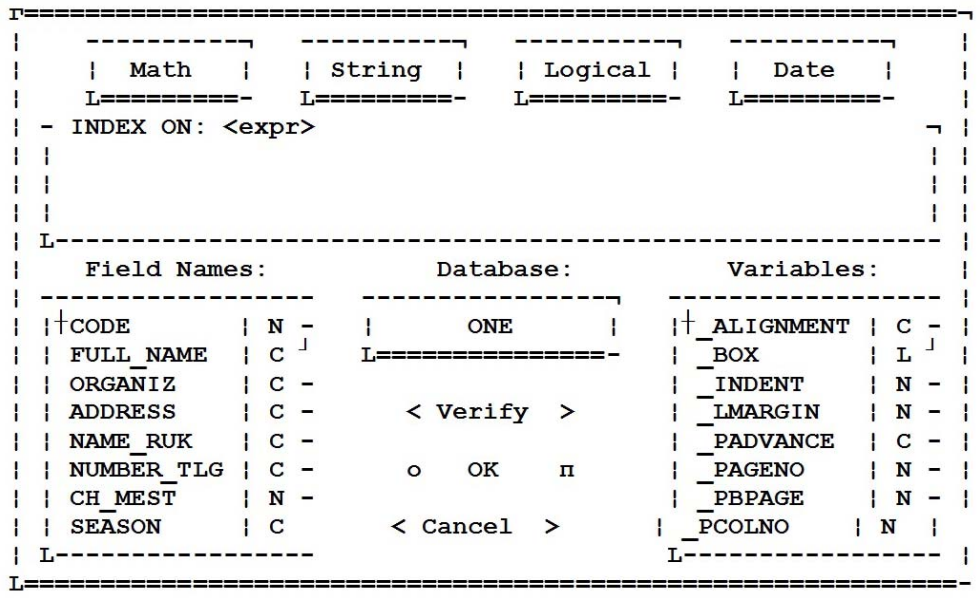


Рис. 4.16 — Меню построителя выражений

Применение выражения арифметического (Math) типа предусматривает упорядочение по физическим номерам записей, что имеет весьма ограниченное применение. Столь же экзотической является и возможность включения в выражения переменных Variables и оператора <For>.

При активизированном переключателе Unigue записи основного файла базы данных (*.dbf) с одинаковыми значениями ключа будут иметь в индексном файле только ссылку на первую запись с таким же значением.

Если сформированное индексное выражение удовлетворяет пользователя, он должен выбрать кнопку <<OK>>, если желает отменить результаты всех действий — кнопку <Cancel>.

МОДИФИКАЦИЯ ИДЕКСНОГО КЛЮЧА (ВЫРАЖЕНИЯ) для уже созданного индексного файла возможна после выбора переключателя <Modify> в окне установки (после Set up) (рис. 4.11). Появляется диалог формирования индекса (рис. 4.15). Необходимо внести изменения в индексное выражение, приведенное в текстовом блоке, либо удалить выражение и создать новое.

Задание № 1 лабораторной работы № 1

1. Внимательно изучить указания.
2. Выбрать вариант (предметную область) по общим правилам, проанализировать предлагаемую структуру таблиц, требуемые индексы и запросы.
3. Создать структуру таблиц, придерживаясь следующих требований:
   - Типы полей следует выбирать в соответствии с семантикой (смыслом) данных, а не типами, устанавливаемыми системой по умолчанию.
   - Имена файлов БД и полей файла БД не должны представлять собой неудобочитаемые и непонятные идентификаторы.
   - Имена файлов БД и полей файла БД по возможности должны представлять собой англоязычные эквиваленты соответствующих русскоязычных понятий.
   - Максимальную длину полей выбирать в соответствии с допустимыми значениями полей заданной предметной области, а не значениями, устанавливаемыми системой по умолчанию.
4. Заполнить файлы БД данными, придерживаясь следующих требований
   - Количество записей в родительской таблице — не менее 15, дочерней — не менее 30.
   - Конкретные значения полей должны быть максимально приближены к реальным. Для этого можно привлекать открытые источники. Допускается программное генерирование значений полей (например, ID, Номер паспорта, Сумма и т.д.) или конвертация данных из внешних файлов с аналогичной структурой. Фривольного содержания не допускать.
   - Каждой строке родительской таблицы должно соответствовать 0, 1 или более строк дочерней.
   - Содержание данных должно гарантированно обеспечивать положительный результат каждого запроса при определенных значениях критерия поиска.
5. Создать индексный файл CDX для заданного в варианте набора условий индексов. Каждый простой индекс внутри CDX обозначить тегом (для последующего обращения к нему).

6. Написать отчет, в котором описать ход работы.
7. На проверку присылать файлы отчета, БД и индексные.

Задание 2. Разработка программ локализации и поиска записей

4.1.10 Разработка программных файлов.

Программные файлы (программный файл — это обычный текстовый файл, каждая строка которого представлена командой FoxPro) можно создавать или модифицировать с помощью встроенного текстового редактора, вызываемого командой:

MODIFY COMMAND [<file>]
[NOEDIT] [NOWAIT]
[RANGE <expN1>, <expN2>]
[WINDOW <window name1>]
[SAVE]

При создании программного файла по умолчанию присваивается расширение .prg. Если имя файла не указано, при выполнении команды создается файл с именем UNTITLED.prg, который в дальнейшем может быть сохранен под другим именем с помощью опции (пункта) меню Save as. . . главного меню FILE.

Опция NOEDIT запрещает модификацию текста программного файла в окне редактирования, допускается только просмотр файла.

Опция NOWAIT позволяет продолжать выполнение программы после открытия окна редактирования (если опция не указана — выполнение программы приостанавливается до закрытия окна).

Опция RANGE позволяет открыть окно редактирования, в котором указанный диапазон символов (<expN1>, <expN2>) уже будет выбран (выделен). Если диапазон не выбирается, курсор остается в позиции <expN1>.

Опция WINDOW <имя окна> — используется для указания имени окна, которое FoxPro будет использовать для редактирования.

Опция SAVE позволяет сохранить текст на экране после выхода из окна редактирования. Опция доступна только внутри программы.

В FoxPro можно использовать модульный принцип построения прикладных программ, когда законченный прием работы с данными (например, поиск информации по определенному условию или организация меню и т.д.) оформляется как процедура. Поэтому программа обычно состоит из главной процедуры и вызываемых из нее процедур, например:

```
< Прикладная программа>  && Начало главной процедуры
DO Proc1
DO Proc2
DO Proc3
RETURN     && Конец главной процедуры
PROCEDURE Proc1  && Начало Proc1
```

```
<команды>
RETURN     && Конец Proc1
. . .
```

Таким образом процедура представляет собой программный модуль, который выполняет определенную задачу. После ее выполнения управление передается в вызвавшую ее главную процедуру.

Процедура так же, как и программа, вызывается и выполняется командой:

DO <имя программы>.

В любой процедуре можно вызвать другие процедуры, формируя, таким образом, вложенные структуры процедур.

Следует, однако, избегать рекурсивного вызова процедур, т. е. вызова из процедуры этой же процедуры. Процедуры можно помещать вместе с главной процедурой или в отдельный файл процедур.

Размещение процедур в программном файле обеспечивает быстрое их выполнение, т.к. не требуется открывать и закрывать процедурный файл и считывать процедуры в оперативную память для выполнения.

В то же время процедуры, которые выполняются только один раз, занимают оперативную память в течение всего времени выполнения прикладной программы. Поэтому целесообразно размещать часто выполняемые процедуры в головной программе, а редко вызываемые — в процедурном файле.

Для доступа к процедурам, размещенным в процедурном файле, необходимо в головной программе использовать команду:

SET PROCEDURE TO <имя процедурного файла>

Каждая процедура, размещаемая в файле процедур либо в программном файле, должна начинаться с команды:

PROCEDURE <имя процедуры>

и завершаться командой:

RETURN TO MASTER

либо просто

RETURN TO <имя прцедуры>

Опция TO MASTER выполняет возврат в программу высшего уровня (в DOS либо в систему) в зависимости от того, откуда был осуществлен вызов процедуры.

Опция TO<имя процедуры> — возвращает управление в активную процедуру с указанным именем.

4.1.11 Локализация и поиск данных.

При работе с записями файлов базы данных часто бывает полезным использовать команды поиска нужных записей для вывода на экран либо принтер не всей информации, а той, которая удовлетворяет некоторому заданному значению. Ввод значений переменных с клавиатуры в процессе

работы прикладной программы производится командами ACCEPT и INPUT (в алгоритмах поиска и локализации данных со значениями этих переменных будут сравниваться значения данных в файле).

ACCEPT применяется для ввода строковых значений, а **INPUT** — числовых и логических. Обе команды позволяют вводить текст подсказки, например:

ACCEPT 'ВВЕДИТЕ СЛОВО' TO x && запомнить строку в переменную x,

INPUT 'введите число' TO y && ввести число в переменную y;

при этом при вводе строки командой INPUT она должна заключаться в кавычки или апострофы.

Команды поиска записей

КОМАНДА LOCATE

Механизм действия этой команды при поиске записей таков, что последовательно просматриваются все записи до нахождения записи, удовлетворяющей условию. Таким образом, с помощью этой команды можно отыскать ***первую запись***, удовлетворяющую условию поиска, для дальнейшего просмотра записей с целью нахождения всех записей, удовлетворяющих условию поиска эта команда должна использоваться вместе с командой CONTINUE, назначение которой состоит в продолжении поиска записей, удовлетворяющих поставленному условию.

• LOCATE FOR <условие > [<границы>] [WHILE<условие>]

В случае, если границы и WHILE — условие отсутствуют, поиск ведется во всем файле, начиная с первой записи.

При успешном поиске указатель записей устанавливается на найденную запись, функция RECNO() равна номеру этой записи, а функция FOUND(), оценивающая результат поиска, возвращает значение .T. («ИСТИНА»).

При неудачном поиске функция RECNO() равна числу записей в базе плюс одна, FOUND() =.F.

Примеры использования этих команд.

Использование команды LOCATE

```
*Locate.prg
 *поиск с помощью LOCATE
 SET TALK OFF
 @ 0,0 Clear
 ACCEPT 'ВВЕДИТЕ НАЗВАНИЕ' TO otv
 USE sk.dbf     &&  Активизируется файл данных sk.dbf, в котором есть
                    поле  kod
 LOCATE for kod=otv
 * предыдущая  строчка  содержит условие сравнения с переменной,
```

```
* которая введена с клавиатуры пользователем
IF EOF()          &&  Если конец файла
? otv+' нет в файле '
ELSE
? kod+STR(power,4,2)   &&  Вывести значения поля kod  и
                                соответствующее ему  значение поля power
ENDIF
SET TALK ON
```

Продолжение поиска

Использование конструкции LOCATE-CONTINUE

```
*  Поиск с помощью конструкции LOCATE  CONTINUE
   ACCEPT 'ВВЕДИТЕ НАЗВАНИЕ ' to otv
   USE sk.dbf
   LOCATE for kod=otv
   DO WHILE FOUND()
   ?kod+STR(power,4,2)
   CONTINUE
   ENDDO
```

Поиск с помощью SEEK

Этот прием рекомендуется использовать при большом объеме файла, т.к. применение этой команды обеспечивает ускоренный поиск.

• SEEK <выражение>

Команда применяет специальный алгоритм ускоренного поиска, в котором база просматривается не сплошь, а в соответствии с информацией, содержащейся в индексе, т. е. для ее использования необходимо использовать индексный файл. Команда SEEK разыскивает только одну первую запись, в которой в индексном поле наблюдается <выражение>. Пример поиска данных с использованием команды Seek

```
   SET TALK OFF
   @0,0 clear
   ACCEPT 'ВВЕДИТЕ НАЗВАНИЕ '  TO otv
   USE  sk  INDEX skkod
   * при использовании команды SEEK файл должен быть проиндексирован
   SEEK  otv  && поиск записи, совпадающей с данными введенными как otv
   IF EOF()   &&  если обнаружен конец файла
   ? otv+'нет в файле'  && вывести строку   значение  otv нет в файле
   ELSE
   ? kod + STR(power,4,2) && str – функция преобразования числового поля в
                                символьное
   ENDIF
   SET TALK ON
```

Поиск с помощью команды SET FILTER

Данная конструкция осуществляет фильтрацию данных, по средствам установки FOR-условия для всех без исключения команд обработки данных. (Например, команда SET FILTER TO fam' Ив'.) Команда SET FILTER действует исключительно на ту базу, которая открыта и активна в данный момент.

SET FILTER TO без параметра снимает все ограничения на предъявление записей из текущей базы данных.

Установление фильтра имеет особенность: он начинает действовать только в случае, если после команды SET FILTER TO <условие> произведено хоть какое-то перемещение указателя записей в файле базы данных.

```
ACCEPT 'ВВЕДИТЕ НАЗВАНИЕ ' TO otv
USE sk.dbf
SET FILTER TO kod=otv
GOTO TOP   && Здесь произведено перемещение по базе.
DO WHILE .NOT.EOF()
? kod + STR(power,4,2)  && Str-функция преобразования данных типа
                                    число в символьный тип
SKIP
ENDDO
```

Результаты работы последних двух программ одинаковые. Однако механизм поиска записей различный. В первой программе команда LOCATE последовательно пробегает записи с начала файла до нахождения записи, которая удовлетворяет условию.

Команда CONTINUE также последовательно перебирает последующие записи.

Во второй программе команда GOTO TOP, активизируя фильтр, сразу устанавливает указатель на первую запись, удовлетворяющую условию фильтрации, команда SKIP пробегает только отфильтрованные записи.

Задание № 2 лабораторной работы № 1

1. В соответствии с выбранным вариантом реализовать запросы, используя индексный способ поиска. Каждый запрос оформить в виде отдельного программного файла.
2. Запустить запросы на выполнение. Для каждого запроса вводить различные значения критериев, которые обеспечивали бы положительный и отрицательный результаты поиска. В случае отрицательного результата должно выводиться соответствующее сообщение.

3. Написать отчет, демонстрирующий работоспособность процедур поиска в соответствии с пунктом 2.
4. На проверку необходимо отправить отчет, программные файлы, файлы БД и индексные.

4.2 Лабораторная работа № 2. Использование генератора приложений

В СУБД FoxPro имеются развитые генераторы приложений, позволяющие быстро создавать, почти не прибегая к непосредственному программированию, некоторые заготовки. Это генераторы отчетов, экранов и меню. Здесь программист во многом освобожден от рутинного и утомительного труда по расчету положений видимых объектов (текстов, полей, рамок, меню и др.) на экране. Все проектирование сводится к физическому размещению нужных элементов в специальном окне проектирования — планшете, облик которого полностью соответствует будущему виду экрана.

В FoxPro для облегчения создания собственного многоуровневого меню предусмотрен генератор меню. Для запуска генератора воспользуйтесь пунктом NEW из меню FILE. В появившемся диалоговом окне выберите пункт MENU и на экране появится пустой планшет (рис. 4.17). Второй способ запуска генератора меню — написать в командном окне команду:

CREATE MENU <имя MNX файла> или
MODIFY MENU <имя MNX файла>

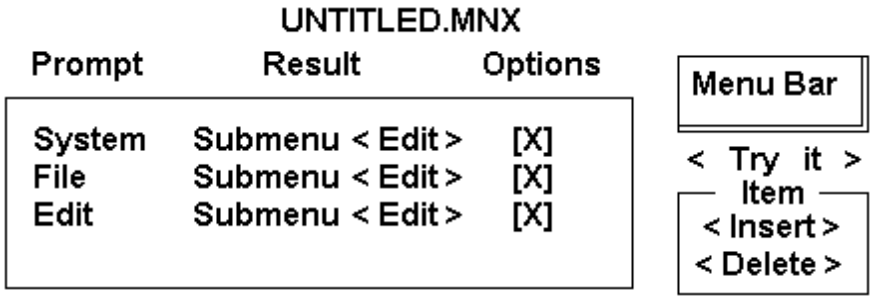


Рис. 4.17

Рабочая часть планшета имеет четыре столбца:

В столбец Prompt заносятся пункты-приглашения создаваемых меню. При этом для первого уровня меню генерируется команда вида DEFINE PAD <имя PAD пункта> OF _MSYSMENU PROMPT <приглашение>

В столбец Result указываются действия, которые вызовет выбор пункта. Выбрать тип действия можно с помощью всплывающего меню. Для его вызова наведите курсор на надпись SUBMENU и нажмите <ВВОД>. Появится меню, состоящее из четырех пунктов, каждый из которых определяет тип действия. Доступны следующие типы действий:

Command позволяет задать некоторую команду, которая будет выполняться при выборе данного пункта меню. Если это пункт меню первого уровня, то в меню-программу добавляется команда вида
ON SELECTION PAD <имя PAD пункта> OF _MSYSMENU <команда>

Или для POPUP меню следующих уровней

ON SELECTION BAR <номер пункта POPUP меню> OF <имя POPUP меню> <команда>

Submenu дает возможность вставить вспомогательное POPUP меню. В программу для горизонтального меню добавляется команда

ON PAD <имя PAD пункта> OF _MSYSMENU ACTIVATE POPUP <имя POPUP меню>

Или для пункта POPUP меню команда ON BAR <номер пункта> OF <имя POPUP меню> ACTIVATE POPUP <имя POPUP меню>

Pad name позволяет задать собственное, а не генерируемое автоматически имя PAD пункта в команде вида

DEFINE PAD <имя PAD пункта> OF _MSYSMENU PROMPT <приглашение>

Здесь подразумевается имя одного из PAD пунктов системного меню, которое вы хотите использовать в конструируемом меню. Каждый элемент системного меню имеет свое специальное имя, которое можно выяснить, например, воспользовавшись пунктом Quick Menu из меню MENU.

Bar # (отсутствует в меню первого уровня) позволяет задать в качестве строки создаваемого POPUP меню элемента одного из системных POPUP меню.

Proc. позволяет задать процедуру, которая закрепляется за соответствующим пунктом меню. Для горизонтального меню к программе добавляется команда

ON SELECTION PAD <имя PAD пункта> OF _MSYSMENU DO <имя процедуры> IN <имя программы, содержащей процедуру>

Для POPUP меню добавляется команда вида

ON SELECTION BAR <номер пункта> OF <имя POPUP меню> DO <имя процедуры> IN <имя программы, содержащей процедуру>

Здесь <имя процедуры> — уникальное имя, назаначаемое генератором меню. Сама процедура может быть написана непосредственно тут же. В зависимости от того, какой объект задан, в этом же столбце (находящемся чуть левее POPUP меню) откроется доступ к их созданию/редактированию через кнопку CREATE/EDIT, либо, если выбран пункт Command, — появится возможность сделать непосредственный ввод команды.

Столбец OPTIONS открывает доступ к окну, показанному на рис. 4.18.

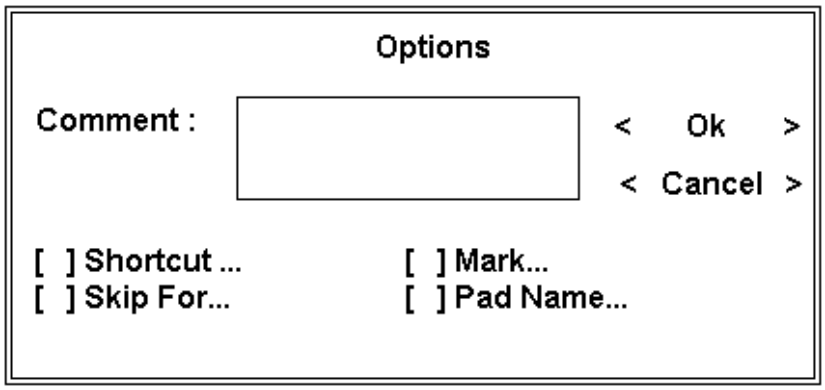


Рис. 4.18

Здесь можно установить следующие опции для элементов меню:

Shortcut — назначение клавиш быстрого вызова данного пункта меню.

Mark — указание символов пометки выбранных пунктов.

Skip For — условие, исключающее доступ к данному пункту.

Pad Name/Bar # — собственное имя заголовка, номер строки меню.

При разработке меню удобнее задавать собственные имена пунктов, а не предлагаемые генератором.

В правом верхнем углу планшета отображено скрытое POPUP меню, где показано имя текущего уровня меню. Там же можно выбрать нужный уровень из уже заданых.

Остальные кнопки имеют следующий смысл:

Try it — просмотр текущего меню в действии.

Insert — вставка нового пункта меню.

Delete — удаление пункта меню.

Пункт меню MENU

Ниже описаны значения его пунктов.

Пункт **General Options** открывает диалоговое окно, показанное на рис. 4.19.

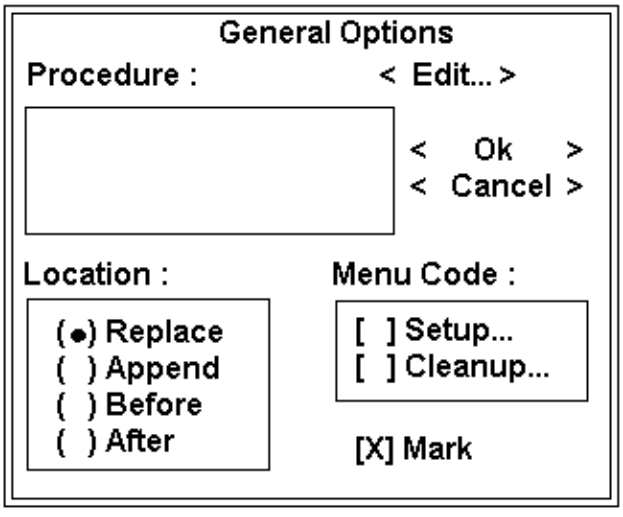


Рис. 4.19

Здесь кнопки Replace и Append в разделе Location устанавливают создаваемое меню вместо системного или дополнительно к нему. Кнопки Before и After — перед или после выбранного пункта системного меню.

В разделе Menu Code можно ввести команды, предваряющие (Setup) или завершающие (Cleanup) установку меню.

Кнопка Mark дает возможность установить вид символа пометки выбранных пунктов.

В окне Procedure через кнопку Edit можно ввести команду, которая будет включена в команду вида ON SELECTION MENU _MSYSMENU <команда>

При выборе пункта Quick Menu для редактирования предлагается системное меню, которое можно приспособить для своих целей.

Генерация программы меню

После окончания формирования меню с помощью пункта GENERATE из меню PROGRAM сгенерируйте файл (он будет иметь расширение MPR), запуск которого приведет к созданию меню. Для вызова этого меню воспользуйтесь командой ACTIVATE MENU <имя меню>.

Задание на лабораторную работу № 2

1. С помощью генератора меню разработать программу, реализующую меню.
2. Меню должно обеспечивать просмотр исходных данных в таблицах с выбором индекса (упорядочения) и вызов реализованных процедур поиска из предыдущей работы, а также выход из системы.
3. На проверку выслать все файлы.

4.3 Лабораторная работа № 3. Разработка экранных форм

Одной из задач систем работы с базами данных — организация дружественного интерфейса, т. е. представление на экране информации в виде удобном для восприятия и изменения.

Для данных целей в СУБД FoxPro существует встроенный генератор экранных форм, позволяющий легко и быстро построить экран для отображения данных, а затем получить программный код построения экрана для его последующей модификации.

4.3.1 Запуск генератора экранов

Запуск генератора экранов можно осуществить двумя способами:

Способ 1: В меню **FILE** выбрать пункт **NEW**, затем в появившемся окне выбрать кнопку **SCREEN** и нажать **Ok**.

Способ 2: В командном окне набрать команду ***CREATE SCREEN*** *<ИМЯ ФАЙЛА>* и нажать **ENTER**.

Для редактирования уже существующего экрана можно использовать меню **FILE** пункт **OPEN** либо выполнить команду ***MODIFY SCREEN*** <Имя SCX-Файла>

После этих действий на экране появится окно генератора экрана (будем называть его планшет), а в главное меню добавится пункт **SCREEN**.

4.3.2 Размещение полей ввода/вывода

Воспользуемся пунктом **QUICK SCREEN** из меню **SCREEN**. После выбора этого пункта на экране появится диалоговое окно (см. рис. 4.20), в котором вы должны выбрать базу данных, для которой надо сформировать экран. Далее появится другое диалоговое окно, через меню которого можно определить исходный вид экрана. Сначала определим способ расположения полей на экране — одной или несколькими колонками. Для этого выберем **By**.

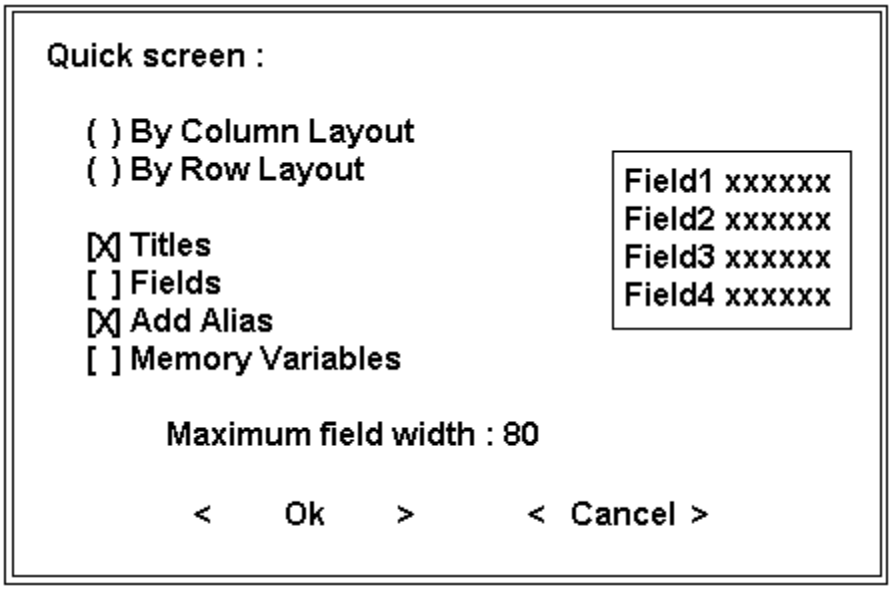


Рис. 4.20

Column или **By Row Layout** соответственно. Ниже расположен ряд кнопок, имеющих следующее назначение:

Кнопка **TITLES** позволяет делать заголовками названия полей.

Кнопка **FIELDS** инициализирует меню выбора полей, используемых на экране.

Кнопка **ADD ALIAS** позволяет делать название полей составными. Это необходимо тогда, когда на экране отображаются поля из нескольких баз данных.

Кнопка **MEMORY VARIABLES** определит создание экрана не для полей базы данных, а для одноименных переменных с префиксом ‘m’ (т. е. если в базе данных есть поле *NAME*, то соответствующая переменная будет иметь имя *m.name*). Это предусмотрено для формирования экранов для временных переменных, значения которых будут копироваться из (или в) базы данных.

С помощью кнопки **FIELD** обратимся к меню выбора полей и выберем нужные поля с помощью кнопок **MOVE** (добавить выделенное в левом окне поле), **ALL** (добавить все поля текущей базы данных), **REMOVE** (убрать поле, при этом оно исчезает из правого окна) и **REMOVE ALL** (убрать все поля). Закончить выбор можно нажав кнопку **Ok** либо **CANCEL**, если результаты ваших манипуляций вас не удовлетворили.

Если вы установили все нужные вам параметры и еще не передумали создавать экран, то нажмите кнопку **Ok**.

На планшете при этом должны появиться выбранные вами поля. Справа от имени каждого поля стоит цифра — порядковый номер. Эти номера соответствуют порядку обхода полей при нажатии клавиши **TAB**. Текущий порядок обхода полей можно изменить с помощью пунктов меню **SCREEN/BRING TO FRONT** и **SCREEN/SEND TO BACK**. Если вручную поочередно выделять поля в желаемом порядке, а затем выбрать один из вышеперечисленных пунктов меню, то поля пронумеруются в порядке их выделения. Следует отметить, что порядок обхода полей не зависит от их расположения на экране и может быть любым. Контрастным цветом выделены области ввода-вывода соответствующих полей.

Теперь можно перемешать названия полей и их области ввода в нужную часть экрана. Для этого надо навести курсор мышки на нужный элемент, нажать левую клавишу мышки и, не отпуская ее перемесить его в нужное место планшета. Можно также добавить комментарий, который будет появляться при выделении поля. Для этого наведите указатель мышки на имя поля и нажмите левую клавишу два раза. В появившемся диалоговом окне нажмите кнопку **COMMENT** и впишите нужные вам комментарии. Также можно сделать на планшете произвольные надписи. Для этого установите курсор на нужное место и наберите нужную вам надпись.

С помощью подпункта **FIELD** из меню **SCREEN** можно добавить новое поле (например, для отображения даты) или отредактировать уже существующее. А с помощью подпункта **BAR** из подменю **SCREEN** можно добавить рамку. Для этого наведите курсор на нужное место, выберите этот пункт из меню и с помощью курсорных клавиш отрегулируйте размер рамки и нажмите **ENTER**. Щелкнув левой клавишей мышки при наведенном на рамку курсоре два раза можно определить тип рамки — двойная или одинарная, или же можно подобрать символ обрамления по своему желанию.

Отредактировать существующее поле также можно, нажав на нем левую клавишу мышки два раза. При этом на экране появляется диалого-

вое окно, в котором можно установить некоторые параметры поля. Примерный вид окна изображен на рис. 4.21. В верхней части окна устанавливается функция поля: **SAY** — поле используется только для отображения информации, **GET** — поле используется для редактирования переменных/полей базы данных, **EDIT** — используется для редактирования полей типа **MEMO**.

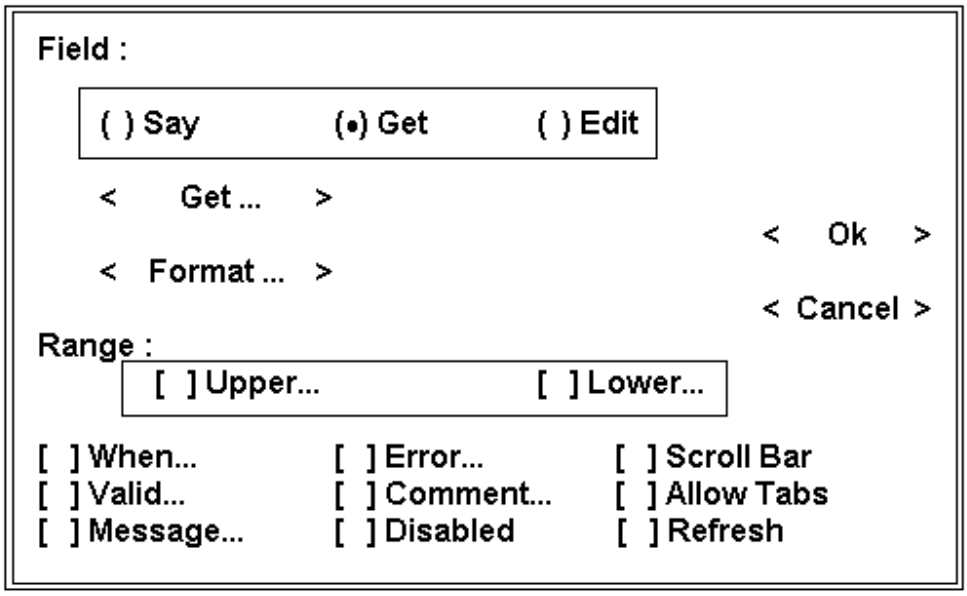


Рис. 4.21

Далее идет окно со строкой вывода для поля типа **SAY**, или стартовое значение для поля типа **GET**. Затем идет строка формата ввода/вывода, которая соответствует шаблону **PICTURE** в операторах **@ , SAY** и **@ , GET**. Наиболее яркий пример использования этого пункта — выбор пола. Например в базе данных есть поле, которое может содержать только два значения: *М* и *Ж*. Все остальные символы вводиться не должны. Для этого в строке формата напишем: **@M М,Ж**.

Еще ниже идут дополнительные опции настройки, соответствующие одноименным параметрам операторов **@ , SAY** и **@ , GET**.

Опция **MESSAGE** используется для задания сообщения, которое появится при попадании курсора на это поле в процессе работы.

4.3.3 Создание кнопок

Еще одна немаловажная деталь — это кнопки. Они существенно упрощают работу с программой. Существует несколько видов кнопок: тригерные кнопки, радиокнопки и кнопки-переключатели. Рассмотрим каждый тип более детально.

Тригерные кнопки (**push buttons**) удобно использовать для движения по записям.

Создадим стандартный набор таких кнопок:

- *Вперед* (продвинуться на запись вперед).

- *Назад* (продвинуться на запись назад).
- *Начало* (переместиться на первую запись).
- *Конец* (переместиться на последнюю запись) и *Выход* (выйти из просмотра базы данных).

Такие кнопки реализуют действия кнопок навигаций и являются удобным и привычным средством передвижения по файлу.

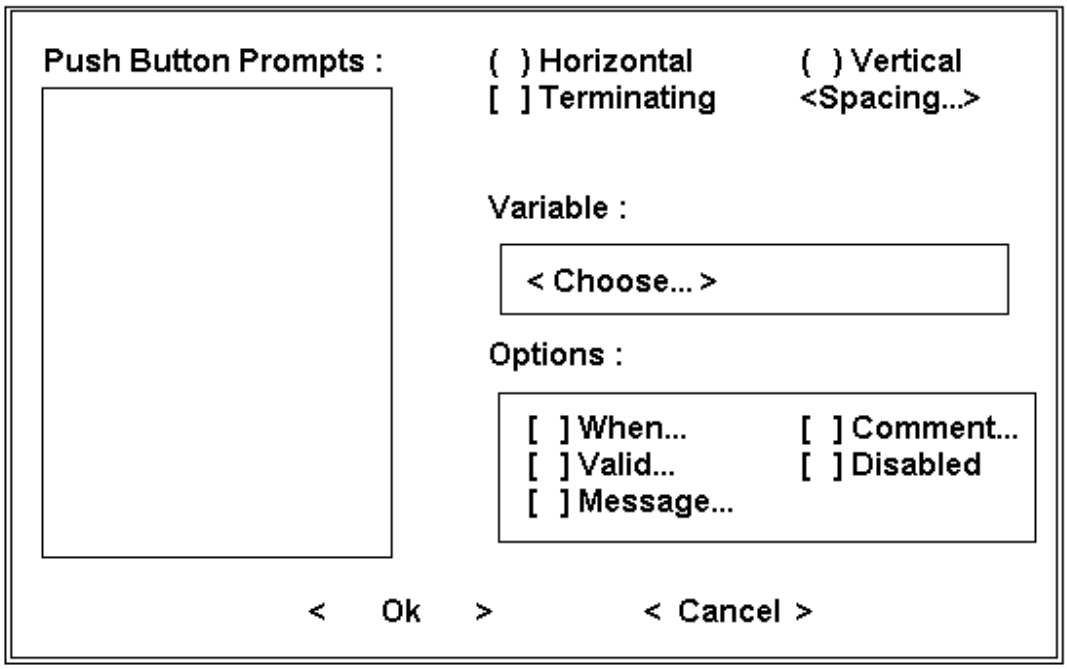


Рис. 4.22

Для реализации этой задачи выберем пункт меню **SCREEN/PUSH BUTTONS**. При этом на экране появится соответствующий диалог настройки кнопок (рис. 4.22). В окне под надписью *'Push Button Prompts:'* перечислите названия кнопок. Затем выберите их расположение с помощью кнопок **HORIZONTAL** и **VERTICAL**, что соответствует горизонтальному либо вертикальному расположению кнопок. С помощью пункта **SPACING** можно задать расстояние между кнопками. В разделе **VARIABLE** (кнопка **<CHOISE...>**) задается переменная, в которую будет записан номер нажатой кнопки. В разделе **OPTIONS** выберите пункт **VALID**, чтобы определить реакцию на выбор меню. Появится диалоговое окно, в котором, выбрав пункт **PROCEDURE**, можно записать текст процедуры. Он может выглядеть примерно так:

```
DO CASE
        CASE k=1
        SKIP 1
        CASE k=2
                SKIP –1
        CASE k=3
```

```
            GO TOP
        CASE k=4
            GO BOTTOM
        CASE k=5
            CLEAR READ
    ENDCASE
    READ CIRCLE
```

В пункте **MESSAGE** можно задать сообщение-подсказку, которая будет появляться, когда будет выбрана какая-либо кнопка. После создания строку с кнопками можно переместить в любое место экрана, как и любой другой объект.

Радиокнопки (**radiobuttons**) используются для установления каких-либо опций (таких, например, как активизация фильтра, пометка записи на удаление и т.п.). Для создания таких кнопок надо выбрать пункт **SCREEN/RADIO BUTTON**.

По аналогии с триггерными кнопками введите названия для радио-кнопок. Общие параметры для них совпадают. Стоит отметить только пункт **INITIAL**, с помощью которого можно установить изначально выбранную кнопку (т. е. кнопку, которая будет выбрана при инициализации).

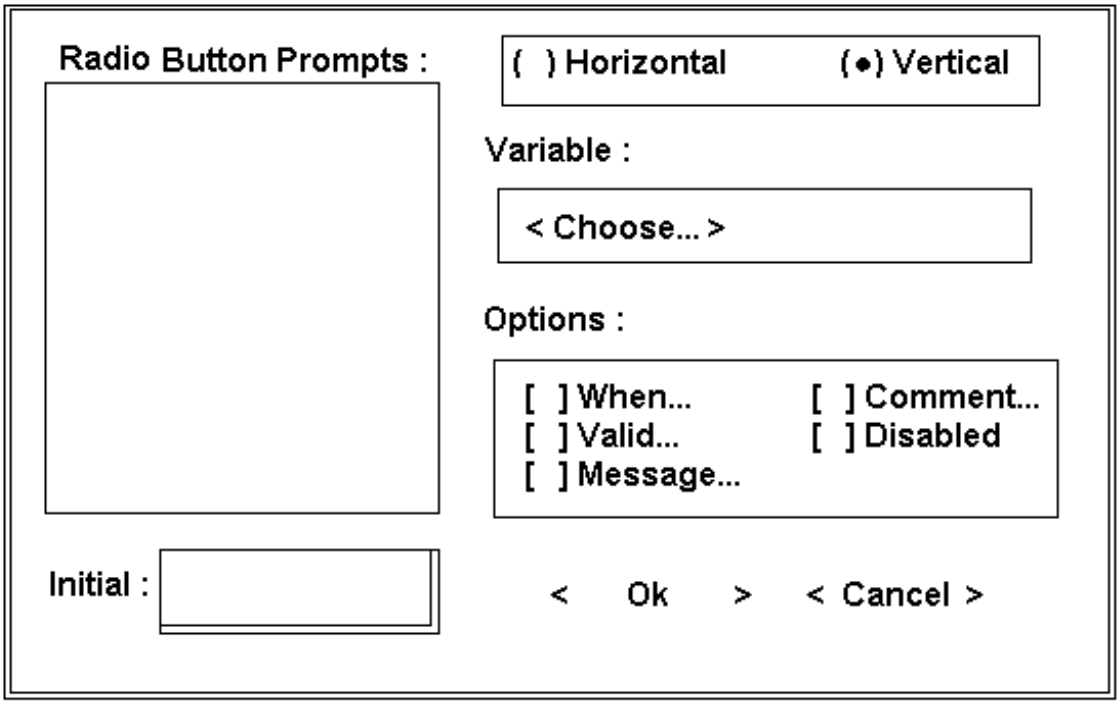


Рис. 4.23

Радиокнопки выглядят примерно так:

(*) Кнопка 1
( ) Кнопка 2
( ) Кнопка 3

Звездочка показывет выбранный пункт, причем из одного блока радиокнопок ‘нажата’ может быть только одна.

Кнопки-переключатели, или **Check box**, это кнопки, которые могут находиться в двух состояниях: включенном и выключенном. Добавить такую кнопку можно, выбрав пункт **SCREEN/CHECK BOX**. На экране должно появиться окно, изображенное на рис. 4.24.

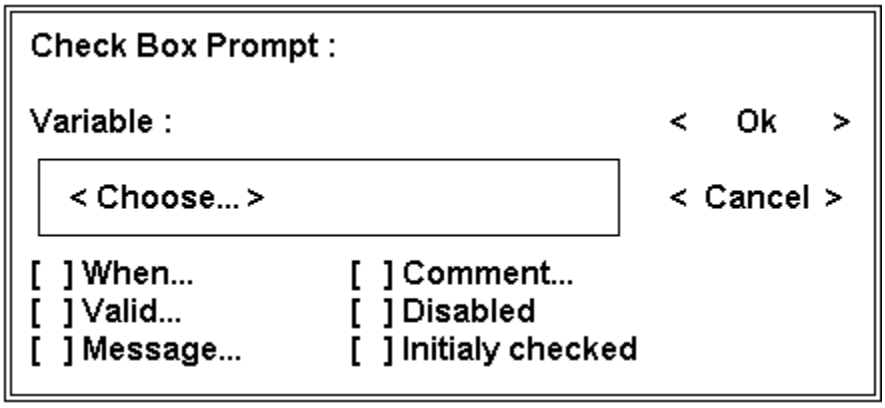


Рис. 4.24

В появившемся диалоговом окне напишите имя кнопки (**Check Box Prompt:**). Остальные параметры аналогичны вышеперечисленным, кроме пункта **Initiality Checked**, который устанавливает, будет ли кнопка включена при инициализации.

4.3.4 Создание POPUP меню

POPUP меню очень удобны, когда в какое-либо поле базы данных необходимо записать значение в соответствии с выбранным из справочника. Например, есть база данных, в которой хранятся марки машин. Нам не обязательно хранить в ней полное название марки автомобиля, достаточно создать справочник, в котором каждому автомобилю будет соответствовать свой номер, который и будет храниться в базе данных. А чтобы пользователю было удобнее, выбор нужной машины можно осуществить через **POPUP** меню, состоящее из названий автомобилей.

Для создания **POPUP** меню необходимо выбрать пункт **SCREEN/POPUP**.

После выбора этого пункта на экране появится окно (см. рис. 4.25), в котором произведем конфигурирование меню.

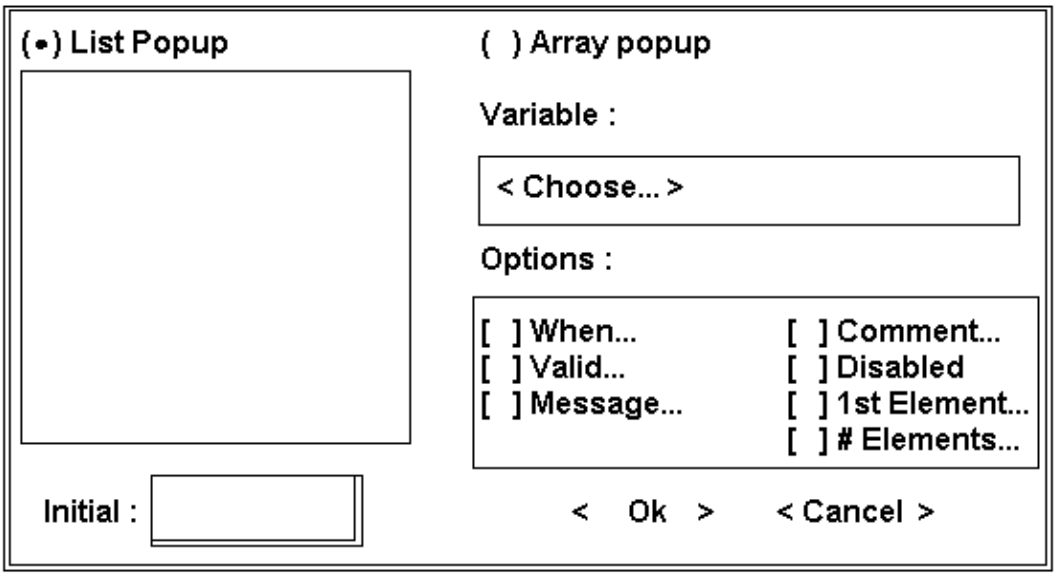


Рис. 4.25

Для начала в разделе **LIST POPUP** введем пункты будущего POPUP меню. Затем, по аналогии с тригерными кнопками, в разделе **<CHOICE...>** зададим переменную, в которую будет записан номер выбранного пункта. С помощью пункта **VALID** можно задать процедуру обработки выбора. В разделе **INITIAL** зададим стартовое значение меню.

4.3.5 Завершающие манипуляции.

В завершении формирования экрана используйте пункт меню **SCREEN/SCREEN LAYOUT** и в появившемся окне (см. рис. 4.26) выберите нужные параметры экрана.

Установите, будет ли сформированный экран окном (выберите кнопку **WINDOW**) либо просто экраном (кнопка **DESKTOP**).

При выборе типа **WINDOW** задайте имя (**NAME**), размер (**SIZE**), заголовок (**TITLE**) и нижнюю надпись (**FOOTER**) окна. С помощью пункта **SCREEN CODE** можно задать команды, которые будут выполнены перед (пункт **SETUP**) и после (пункт **CLEANUP & PROCS**) предьявления сгенерированного экрана. Это могут быть команды очистки экрана, инициализации баз данных, индексов, а также команды настроек среды.

В разделе **ENVERONMENT** предусмотрена возможность сохранения настроек среды.

В результате проделанной работы должен получиться файл с расширением **SCX**, в котором хранится образ экрана, а также файл **SCT**, в котором хранится состояние среды, открытие баз данных и индексов и т.п.

Из полученного экрана можно сделать **SPR**-файл, содержащий программу на FoxPro, в результате выполнения которой на экране появится созданное окно. Для этого в меню **PROGRAM** выберите пункт **GENERATE**.

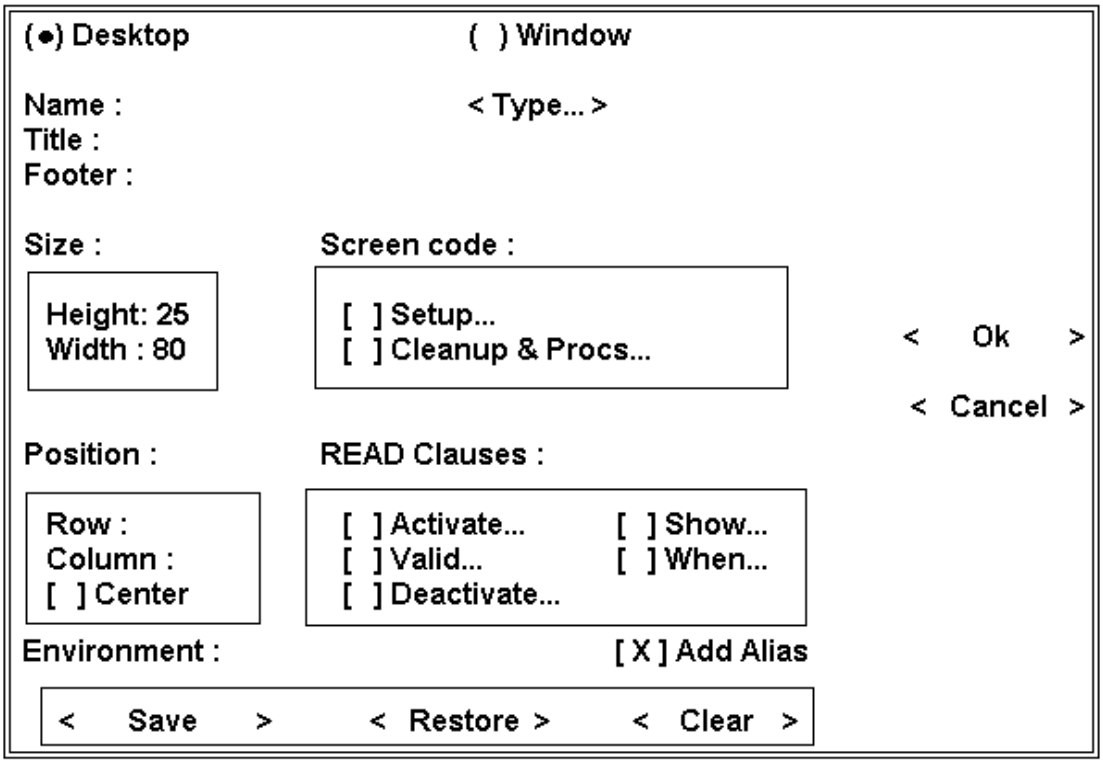


Рис. 4.26

Задание на лабораторную работу № 3

1. С помощью генератора экранов разработать две формы.
2. Основная форма должна быть создана для родительской таблицы, обеспечивать перемещение, ввод записей и вызов дочерней формы посредством кнопок.
3. Дочерняя форма должна в табличном виде отображать все записи дочерней таблицы, соответствующие текущей записи родительской.
4. В отчете описать ход создания форм и продемонстрировать их работоспособность.
5. На проверку выслать файлы шаблонов scx, файл .spr, файлы БД и индексные.

4.4 Лабораторная работа № 4. Одновременная работа с несколькими файлами данных

Реальные базы данных, как правило, состоят из нескольких файлов данных. Часто информацию, хранящуюся в разных файлах, необходимо обрабатывать одновременно. СУБД FoxPro позволяет своими средствами проделывать операции, позволяющие устанавливать связь между базами и объединять все данные в одну задачу. Особое внимание надо обратить на следующие операции:

– SET RELATION

– SET VIEW
– JOIN WITH и др.

Прежде чем познакомиться с работой этих команд, необходимо уяснить, что для работы с несколькими файлами данных они должны быть активированы и помещены в отдельной РАБОЧЕЙ ОБЛАСТИ. Всего рабочих областей может быть 10, и один файл может быть открыт не более чем в одной области.

Выбор/(смена) рабочей области выполняется командой **SELECT.**

Рабочие области можно указывать цифрами 1—10, буквами A—J, именем открытого файла данных либо присваивать рабочей области псевдоним ALIAS-имя. Присвоение рабочей области ALIAS-имени производится при открытии файла командой USE, например:

```
SELECT 2     && выбрать рабочую область 2;
USE spisok INDEX name,group ALIAS f2  && открыть файл в
                                      рабочей области 2 и
                                      присвоить ей псевдоним f2
```

После присвоения файлу данных «псевдонима» (alias) обращение к файлу может происходить по этому имени. Если же в команде USE операнд ALIAS не указывается, то псевдонимом становится само имя базы данных без расширения .dbf.

В каждый момент времени активной может быть одна рабочая область.

При переходе из одной рабочей области в другую (командой SELECT) все файлы в них остаются открытыми. Для файлов данных в каждой рабочей области устанавливается независимый «указатель» текущей записи, а при смене активной рабочей области все «указатели» сохраняют свои значения.

4.4.1 Команда манипулирования данными SET RELATION

Общий формат команды,устанавливающией связь (RELATION) между базами:

SET RELATION TO [<выражение> INTO<имя базы>],

где **<выражение>** — это либо строковое выражение, которое задает значение ключа для поиска в «сыновьей» базе, либо цифровое выражение, которое задает номер записи в «сыновьей» базе. В качестве ключа может использоваться имя поля, по которому осуществляется связь. Понятие «связь» механически можно интерпретировать как «склеивание» по конкретному полю двух и более файлов данных;

<имя> — альтернативное имя сыновьей базы, открытой в другом разделе (другой рабочей области).

Можно указать системе, что при изменении значения указателя записи в текущей базе нужно также изменять указатель записи в базе данных, указаной в операнде INTO. Правило для получения нового значения указателя задано в операнде TO в виде выражения.

Простейшей связью между базами может быть синхронное движение указателя в обеих базах. Для этого достаточно использовать функцию RECNO().

SET RELATION TO RECNO () INTO [<имя базы>]

Например,

SET RELATION TO RECNO() INTO Spisok

указывает системе, что при изменении значения указателя в текущей базе данных, нужно также изменять указатели записи в базе данных, указанной в операнде **INTO**

Такой прием возможен лишь для случая, когда количество записей в базах данных одинаково.

Как видно, в приведенной команде отсутствует ссылка [<выражение>], так как «склеивание « баз происходит последовательно запись за записью.

В общем случае, когда количество записей в базах не обязательно совпадает, «склеивание» баз происходит по общему полю, имя которого и указывается в выражении.

Например, вы имеете две базы данных SPISOK и GRUPA, которые содержат общее поле фамилии студентов NAME. Поэтому лучше, чтобы при переходе в базе SPISOK от одной фамилии к другой в базе GRUPA осуществлялся поиск записи, имеющей такое же содержание поля NAME.

В этом случае мы говорим, что связь между базами осуществляется по полю NAME.

Система требует, чтобы «сыновняя» /т. е. активизированная **не** последней/, база данных была индексирована по этому полю. Значит для получения правильного результата соединения баз надо написать:

```
SELECT 1
USE Grupa INDEX Name
SELECT 2
USE Spisok
SET RELATION TO Name INTO Grupa
DISPLAY ALL FIELDS Nam, Fam,Grupa.Vozrast,Grupa.Group,
```

Как видите, для доступа к полям записей другой (неактивной) рабочей области / в команде DISPLAY / перед именем поля следует указывать имя файла и имя поля(Grupa.Vozrast) или, если рабочие области вместо числового обозначения (1..10) имеют символьные алиасы (a..z), то можно доступ указать <имя рабочей области>.<имя поля> (a.Vozrast).

Команда SET RELATION позволяет установить взаимосвязь между несколькими базами и просмотреть интересующие поля, которые указываются в команде DISPLAY.

В нашем примере поле Name просматривается из файла Spisok, а Vozrast и Group из файла Grupa,обращение к которому происходит с помощью указания номера рабочей области, в которой он активизирован.

Но команда RELATION создает временный экран, и не запоминает связанные данные как новый файл данных. (Можно, конечно, запомнить эти данные, если приведенные команды «склеивания» баз отработать как программный файл).

4.4.2 Команда слияния файлов JOIN WITH

Это более сложная команда манипулирования данными, она дает возможность создавать новую базу данных, которая является соединением двух имеющихся баз или их частей, т. е. она осуществляет слияние «по горизонтали».

JOIN WITH <альтернативное имя/ область> TO<имя нового файла> FOR<условие>[FIELDS<список полей>]

С ее помощью можно объединить путем слияния две открытые базы данных из разных рабочих областей. При этом одна из них является активной, а альтернативное имя другой указывается в операнде WITH .

Если список полей не задан, то вначале будут включены все поля первой базы, за исключением memo-полей, которые нельзя использовать при слиянии, а затем все поля другой.

Условие FOR используется для того, чтобы при слиянии можно было варьировать условием включения записей.

Порядок работы этой команды такой:

1. Выбирается запись из активной базы данных. За ней включаются все записи, которые удовлетворяют условию FOR из альтернативной базы данных.

2. После этого извлекается следующая запись из активной базы данных и процесс повторяется.

Таким образом, после операнда FOR указывается так называемое поле подбора.

Пример программы:

Пусть требуется из файлов KADR.DBF и BRIG1.DBF создать третий файл, который будет содержать поля FAM (фамилия) и TAB (табель) из файла KADR.DBF и поле VIR (выработка) из BRIGI.DBF. Новый файл назван KADRVIR.DBF.

```
SELECT a
USE  KADR.DBF
```

```
SELECT b
USE BRIG1
JOIN WITH a TO KADRVIR FOR tab = a.tab FIELDS  a.fam, tab, vir
```

Основное отличие этих команд состоит в том, что в результате работы этой команды создается новый файл данных (в нашем случае под именем KADRVIR), который содержит поля, указанные после операнда FOR, либо все поля связываемых баз, если этот операнд пуст.
Вновь созданный файл имеет расширение .dbf.

Задание на лабораторную работу № 4

1. Изучить методические указания к лабораторной работе.
2. В командном режиме (т. е. с командной строки) осуществить слияние двух файлов .dbf с помощью команды JOIN WITH.
3. Осуществить слияние двух файлов данных с помощью команды SET RELATION и SET SKIP и оформить последовательность команд в виде программного файла.
4. В отчете продемонстрировать исходные данные и результат слияния обоими способами.
5. На проверку выслать отчет, необходимые файлы БД и индексы, программный и результирующий файл БД.

5 ТЕКСТОВАЯ КОНТРОЛЬНАЯ РАБОТА

Текстовая контрольная работа содержит практические задания, направленные на закрепление знаний основ языка SQL и выполняется для того же варианта, что и лабораторные работы.

5.1 Задание текстовой контрольной работы

1. Проанализировать предложенную схему и описать (как в варианте), какие таблицы с какими полями будут добавлены. Дополнить схему по крайней мере двумя таблицами. Представить схему таблиц в графическом виде.
2. Привести операторы DDL, реализующие создание таблиц, первичных и внешних ключей, ориентируясь на какую-либо произвольную СУБД. Типы полей должны соответствовать смыслу данных и выбранной СУБД.
3. Сфомулировать (как в варианте) по крайней мере два новых запроса. Каждый запрос должен производить выборку из двух таблиц.
4. Привести операторы SELECT для каждого запроса (в варианте и свои).
5. На проверку прислать отчет с выполненными заданиями.

ПРИЛОЖЕНИЕ А
Варианты предметных областей

ВАРИАНТ № 1. Авиаперевозки

Таблицы:

РЕЙС (ID_рейса, Номер рейса, Код авиакомпании, Аэропорт отправления, Аэропорт назначения, Время вылета, Время прилета, Признак прилета на следующие сутки) — отражает сведения о запланированных сегментах перелета.

ВЫЛЕТ (ID_рейса, Фактичествое время вылета, Фактическое время прилета, Дата вылета).

Если используемая версия СУБД не поддерживает тип Время, то поле Время разделить на Час-Минута.

Индексы:

1. ID_рейса в таблице РЕЙС — уникален.
2. {ID_рейса, Дата вылета} — уникальны.
3. По коду авиакомпании.
4. По аэропорту назначения.
5. По фактической дате вылета в обратном порядке.

Запросы:

1. Найти сведения обо всех рейсах заданной авиакомпании.
2. Найти сведения обо всех рейсах, прибывающих на следующие сутки.
3. Найти сведения обо всех рейсах с продолжительностью полета более заданной.
4. Найти сведения обо всех рейсах, которым не соответствует ни один вылет.

ВАРИАНТ № 2. Поставки

Таблицы:

ПОСТАВЩИК (ID_поставщика, ИНН, Город, Юридический адрес, Телефон, Организационно-правовая форма, Дата регистрации в ЕГРЮЛ, Дата завершения регистрации) — сведения о существующих поставщиках.

ПОСТАВКА (ID_поставщика, Дата поставки, Название товара, Количество, Стоимость) — сведения о поставках поставщиков.

Индексы:

1. ID_поставщика в таблице ПОСТАВЩИК — уникально.
2. {ID_поставщика, Дата поставки} — уникально.
3. ИНН поставщика — уникально.
4. По организационно-правовой форме.
5. По дате регистрации в обратном порядке.

Запросы:

1. Найти сведения обо всех поставщиках с периодом действия регистрации более 3-х лет.

2. Найти сведения о поставщике по заданному номеру телефона.
3. Найти сведения обо всех поставщиках, расположенных в заданном городе на определенной улице.
4. Найти сведения обо всех поставщиках, которые не произвели ни одной поставки.

ВАРИАНТ № 3. Сотрудники

Таблицы:

СОТРУДНИК (ID_сотрудника, Фамилия, Имя, Отчество, Пол, Дата рождения, Должность, Стаж).

Ребенок (ID_сотрудника, Имя ребенка, Пол ребенка, Дата рождения).

Индексы:

1. ID_сотрудника в таблице СОТРУДНИК — уникально.
2. { ID_сотрудника, Имя ребенка } — уникально.
3. По полу сотрудника.
4. По дате рождения ребенка в обратном порядке.
5. По стажу.

Запросы:

1. Найти сведения обо всех сотрудниках старше заданного количества лет.
2. Найти сведения обо всех сотрудниках со стажем менее заданного количества лет.
3. Сведения о бухгалтерах мужского пола.
4. Найти сведения обо всех бездетных сотрудниках.

ВАРИАНТ № 4. Туристическое агентство

Таблицы:

КЛИЕНТ (ID_клиента, Номер паспорта, Серия паспорта, Фамилия, Имя, Отчество, Дата рождения, Пол).

ПУТЕВКА (ID_клиента, Дата выезда, Продолжительность, Стоимость, Страна пребывания).

Индексы:

1. ID_клиента в таблице КЛИЕНТ — уникально.
2. {Номер паспорта, Серия паспорта} — уникально.
3. По продолжительности пребывания в обратном порядке.
4. По дате рождения.
5. По стране пребывания.

Запросы:

1. Найти сведения о клиенте с заданными паспортными данными.
2. Найти сведения обо всех клиентах женского пола старше 60-ти лет.
3. Найти сведения о путевках в заданную страну.
4. Найти сведения о клиентах, которые не приобрели путевку.

ВАРИАНТ № 5. Гостиница

Таблицы:

КЛИЕНТ (ID_клиента, Номер паспорта, Серия паспорта, Фамилия, Имя, Отчество, Дата рождения, Пол, Гражданство).

ЗАСЕЛЕНИЕ (ID_клиента, Дата заселения, Время, Срок пребывания, Номер).

Индексы:

1. ID_клиента в таблице КЛИЕНТ — уникально.
2. {Номер паспорта, Серия паспорта} — уникально.
3. По сроку пребывания.
4. По дате рождения в обратном порядке.
5. По дате и времени заселения.

Запросы:

1. Найти сведения о клиенте с заданными паспортными данными.
2. Найти сведения обо всех иностранных клиентах.
3. Найти сведения о заселениях, произведенных в заданную дату.
4. Найти сведения о клиентах, которые не заселялись ни разу.

ВАРИАНТ № 6. Водоканал

Таблицы:

КЛИЕНТ (ID_клиента, № лицевого счета, Название улицы, Номер дома, Номер квартиры, Количество проживающих).

СЧЕТ (ID_клиента, Расчетный месяц, Расчетный год, Дата выставления, Сумма).

Индексы:

1. ID_клиента в таблице КЛИЕНТ — уникально.
2. { ID_клиента, Расчетный месяц, Расчетный год } — уникально.
3. По сроку дате выставления счета в обратном порядке.
4. По количеству проживающих.
5. По дате и времени заселения.

Запросы:

1. Найти сведения о клиенте с заданными паспортными данными.
2. Найти сведения обо всех клиентах, проживающих на заданной улице с заданным количеством проживающих.
3. Найти сведения о счетах, выставленных в текущем месяце.
4. Найти сведения о клиентах, которым не было выставлено ни одного счета.

ВАРИАНТ № 7. Междугородние перевозки

Таблицы:

ВОДИТЕЛЬ (ID_водителя, Номер паспорта, Серия паспорта, Фамилия, Имя, Отчество, Стаж, Категория).

МАРШРУТ (ID_водителя, Дата выезда, Время выезда, Город назначения).

Индексы:

1. ID_водителя в таблице ВОДИТЕЛЬ — уникально.
2. { ID_водителя, Дата выезда } — уникально.
3. По стажу в обратном порядке.
4. По городу назначения.
5. По дате выезда.

Запросы:

1. Найти сведения о водителях заданной категории со стажем более 10 лет.
2. Найти сведения о маршрутах в заданный город.
3. Найти сведения о маршрутах за текущий месяц.
4. Найти сведения о водителях, которые не были отправлены в маршрут.

ВАРИАНТ № 8. Фильмотека

Таблицы:

ФИЛЬМ (ID_фильма, Название, Режиссер, Жанр, Год выпуска, Продолжительность фильма, Студия, Страна).

КОПИЯ (ID_фильма, Тип носителя, Формат записи, Износ, Дата изготовления).

Индексы:

1. ID_фильма в таблице ФИЛЬМ — уникально.
2. { Название, Студия } — уникально.
3. По году выпуска фильма в обратном порядке.
4. По дате изготовления копии.
5. По типу носителя.

Запросы:

1. Найти сведения о копиях с заданным типом носителя и износом более 80%.
2. Найти сведения о фильмах заданного режиссера.
3. Найти сведения о фильмах, выпущенных за два предыдущих года.
4. Найти сведения о фильмах, для которых не было выпущено копий.

ВАРИАНТ № 9. Кредитование

Таблицы:

КЛИЕНТ (ID_клиента, Номер паспорта Фамилия клиента, Имя клиента, Отчество клиента, Дата рождения, Адрес, Телефон).

ЗАЯВКА (ID_клиента, Дата подачи заявки, Сумма кредита, Процентная ставка, Средняя сумма заработка).

Индексы:

1. ID_клиента в таблице КЛИЕНТ — уникально.
2. { ID_клиента, Дата подачи заявки } — уникально.
3. По сумме кредита в обратном порядке.
4. По дате рождения.
5. По процентной ставке.

Запросы:

1. Найти сведения о клиенте с заданными паспортными данными.
2. Найти сведения обо всех клиентах мужского пола старше 50-ти лет.
3. Найти сведения заявках с заданной процентной ставкой.
4. Найти сведения о клиентах, которые не подали ни одной заявки.

ВАРИАНТ № 10. Транспорт

Таблицы:

ГРАЖДАНИН (ID_гражданина, Номер паспорта, Фамилия, Имя, Отчество, Адрес).

АВТО (ID_гражданина, VIN, Марка авто, Цвет авто, Дата выпуска, Мощность).

Индексы:

1. ID_гражданина в таблице ГРАЖДАНИН — уникально.
2. VIN — уникально.
3. По фамилии гражданина.
4. По дате выпуска авто в обратном порядке.
5. По мощности авто.

Запросы:

1. Найти сведения о гражданине с заданными паспортными данными.
2. Найти сведения обо всех авто заданной марки с мощностью более заданной.
3. Найти сведения обо всех авто с заданным годом выпуска.
4. Найти сведения о гражданах, которым не принадлежит ни одного автомобиля.